THE UNIVERSITY OF NEW MEXICO

ECE 538
Advanced Computer Architecture

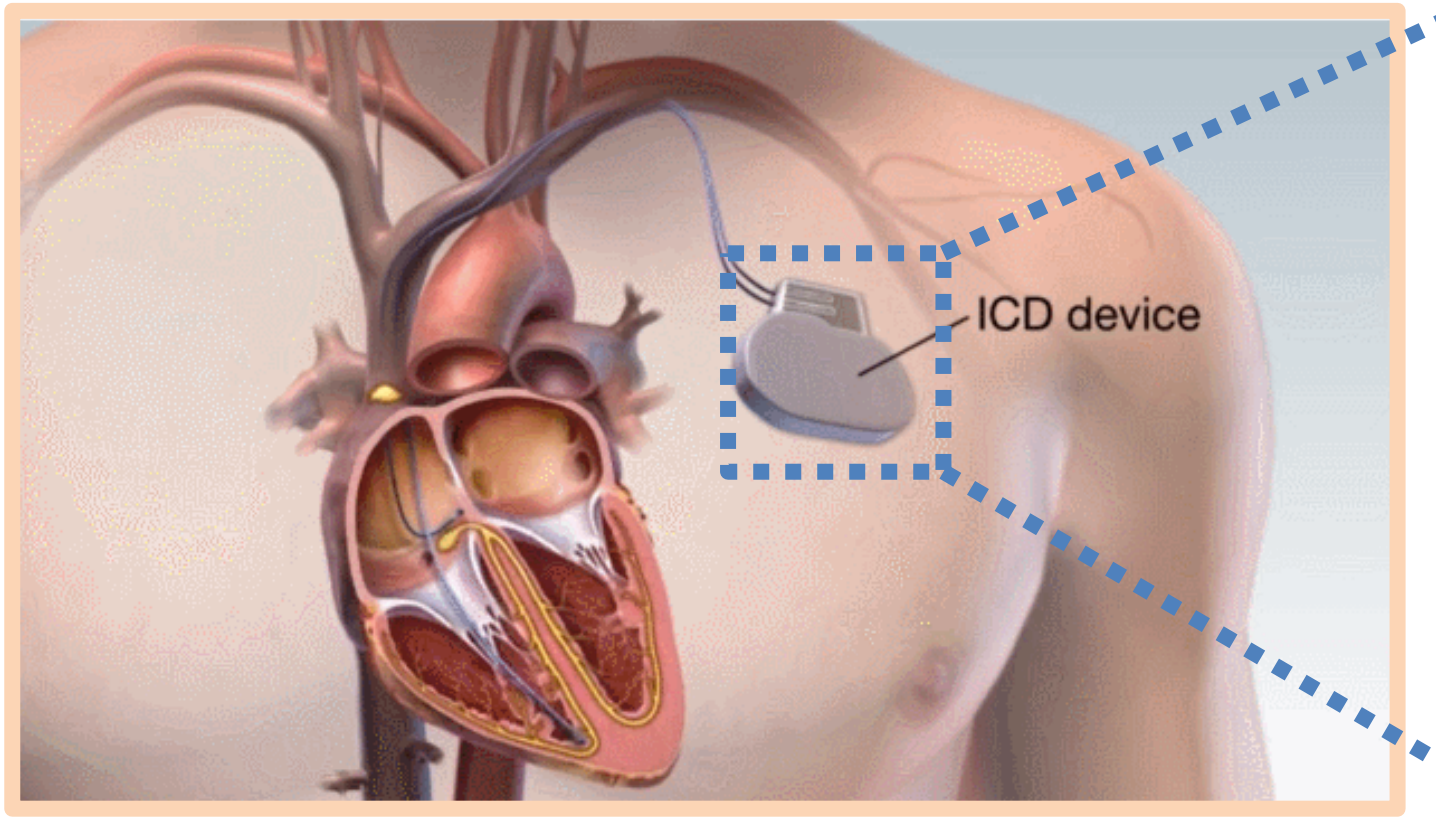
Instructor: Lei Yang

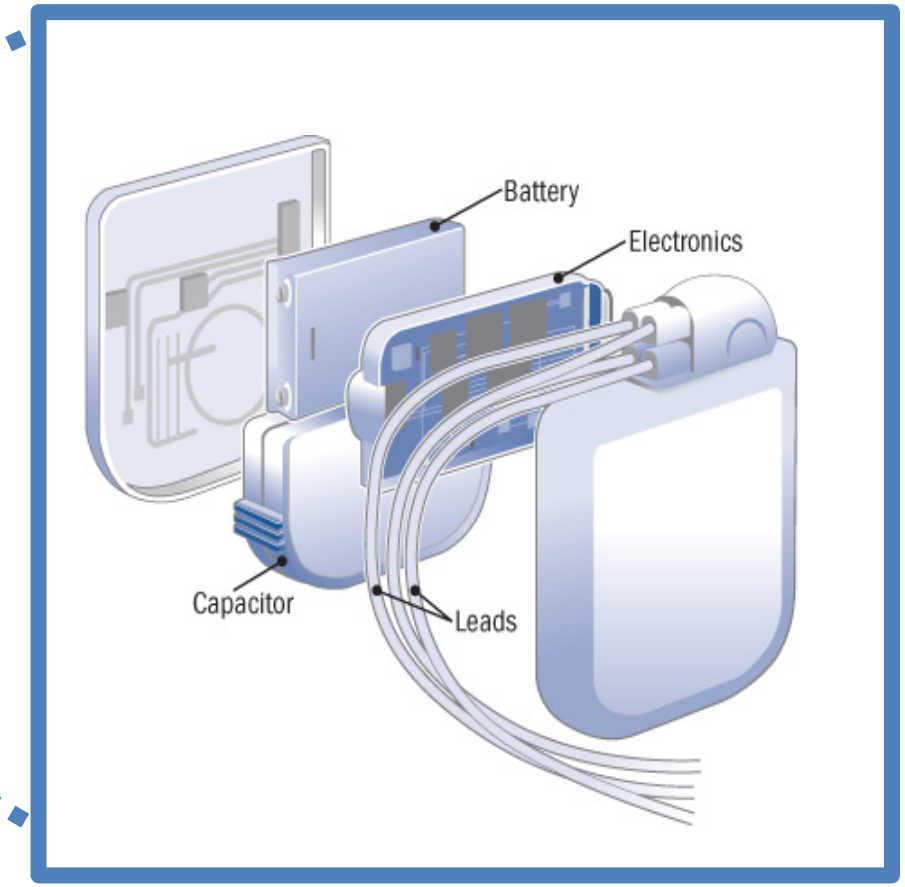
Department of Electrical and Computer Engineering

November 29, 2021

Hardware/Software Co-Exploration of Resource Constrained Edge AI Systems

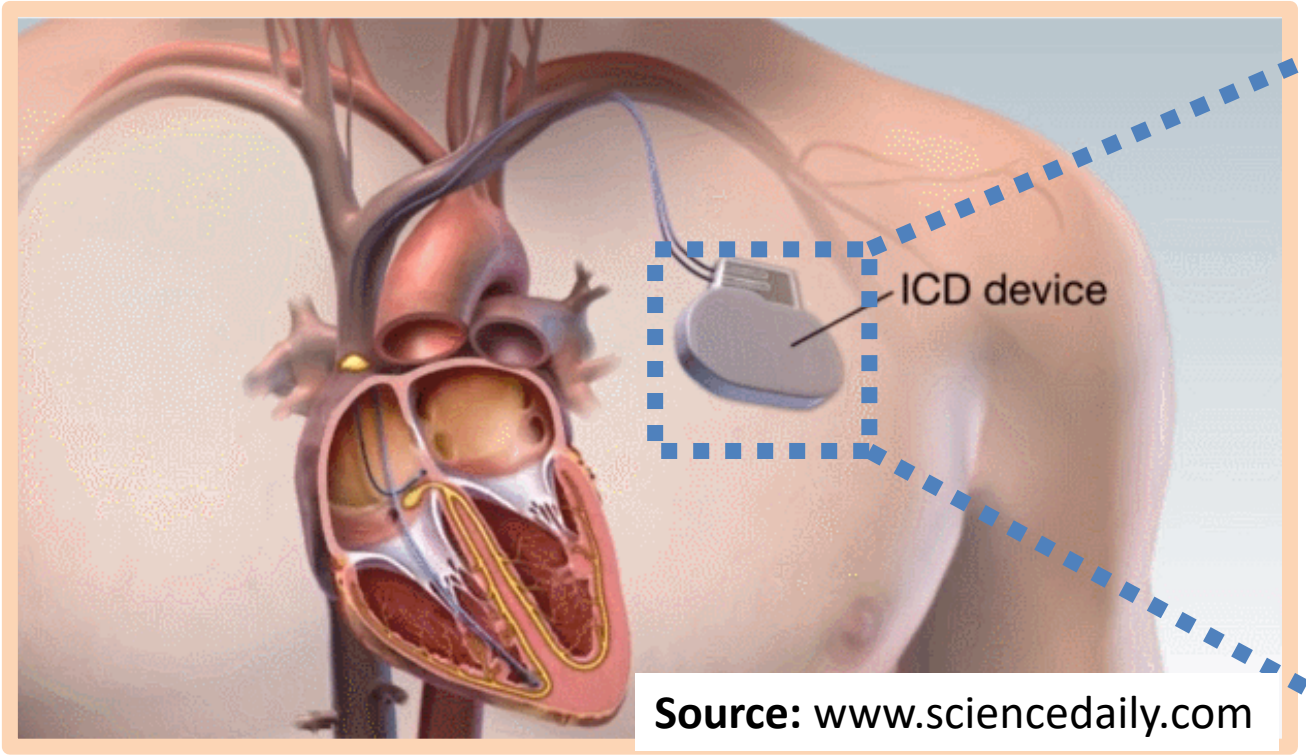
Implantable Cardioverter Defibrillators (ICD)

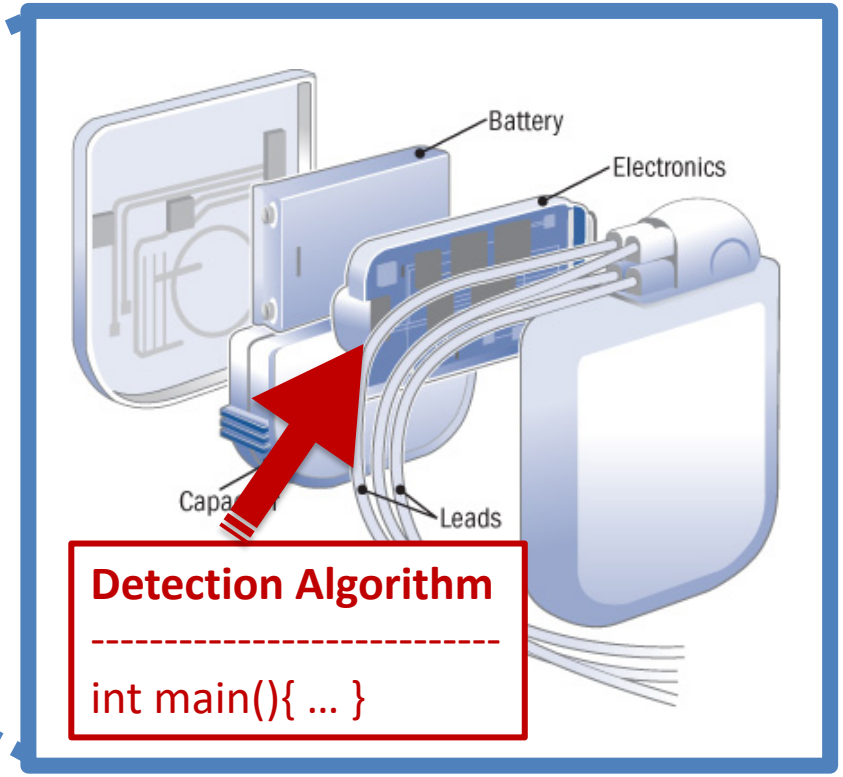




Source: www.sciencedaily.com

Implantable Cardioverter Defibrillators (ICD)





Problem in Existing Detection Approaches:

- Heuristic algorithms are not accurate
- More than **40% shocks** are inappropriate or missing

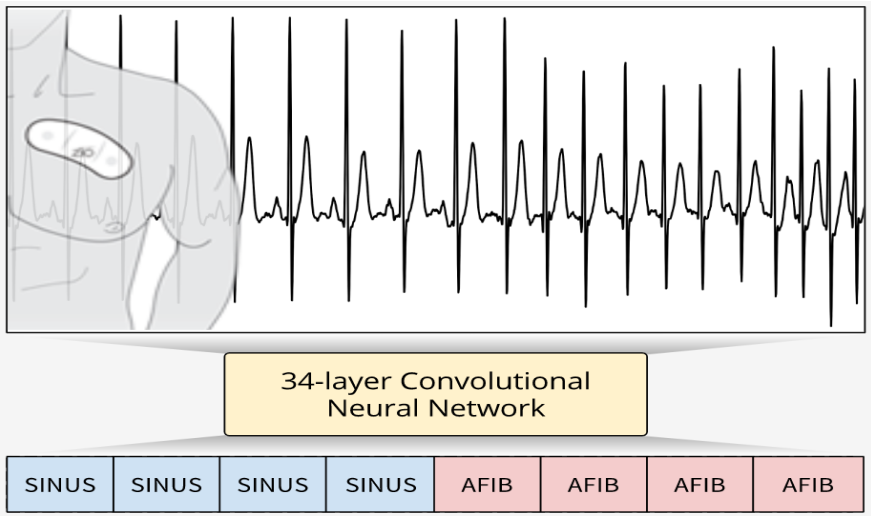
Appropriate Shocks Ratio

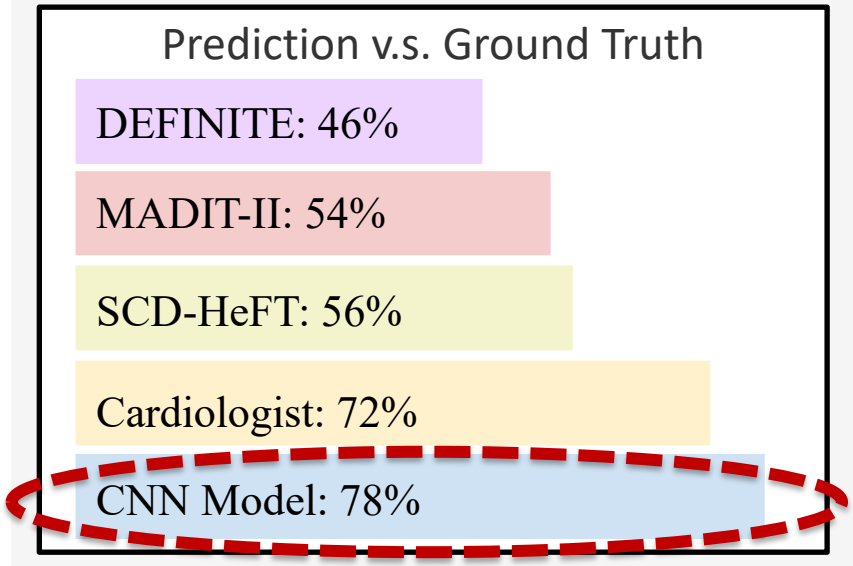
DEFINITE: 46%

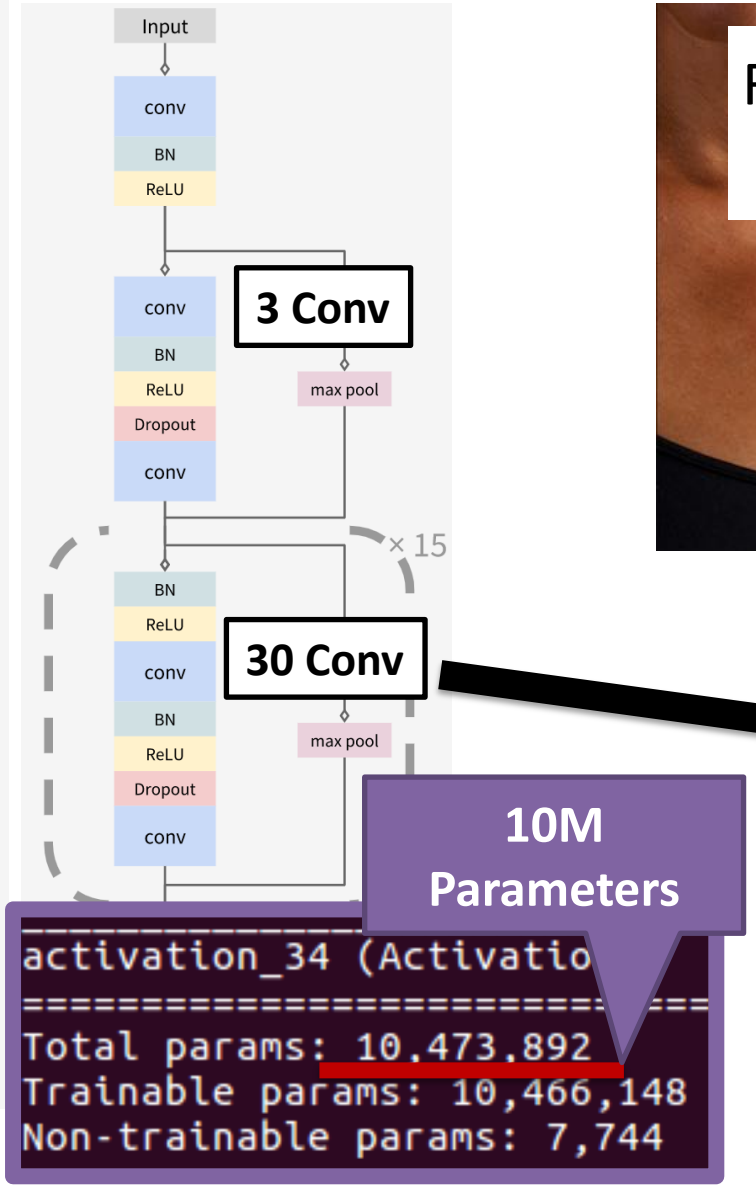
MADIT-II: 54%

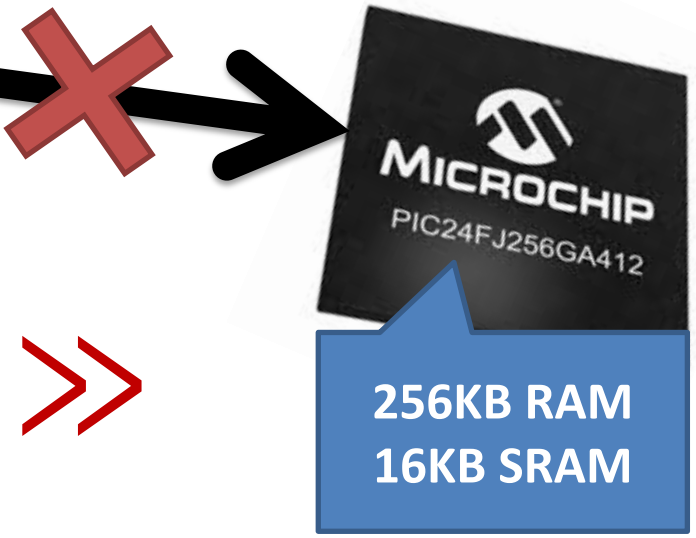
SCD-HeFT: 56%

Machine Learning: Better Than Cardiologist, But …



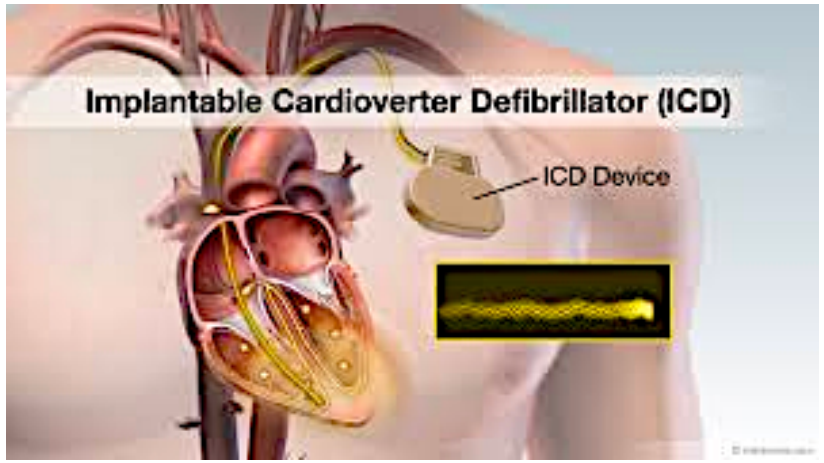






Source: Stanford ML Group (Andrew Ng.)

Put Machine Learning Model in Resource Constrained Edge Devices



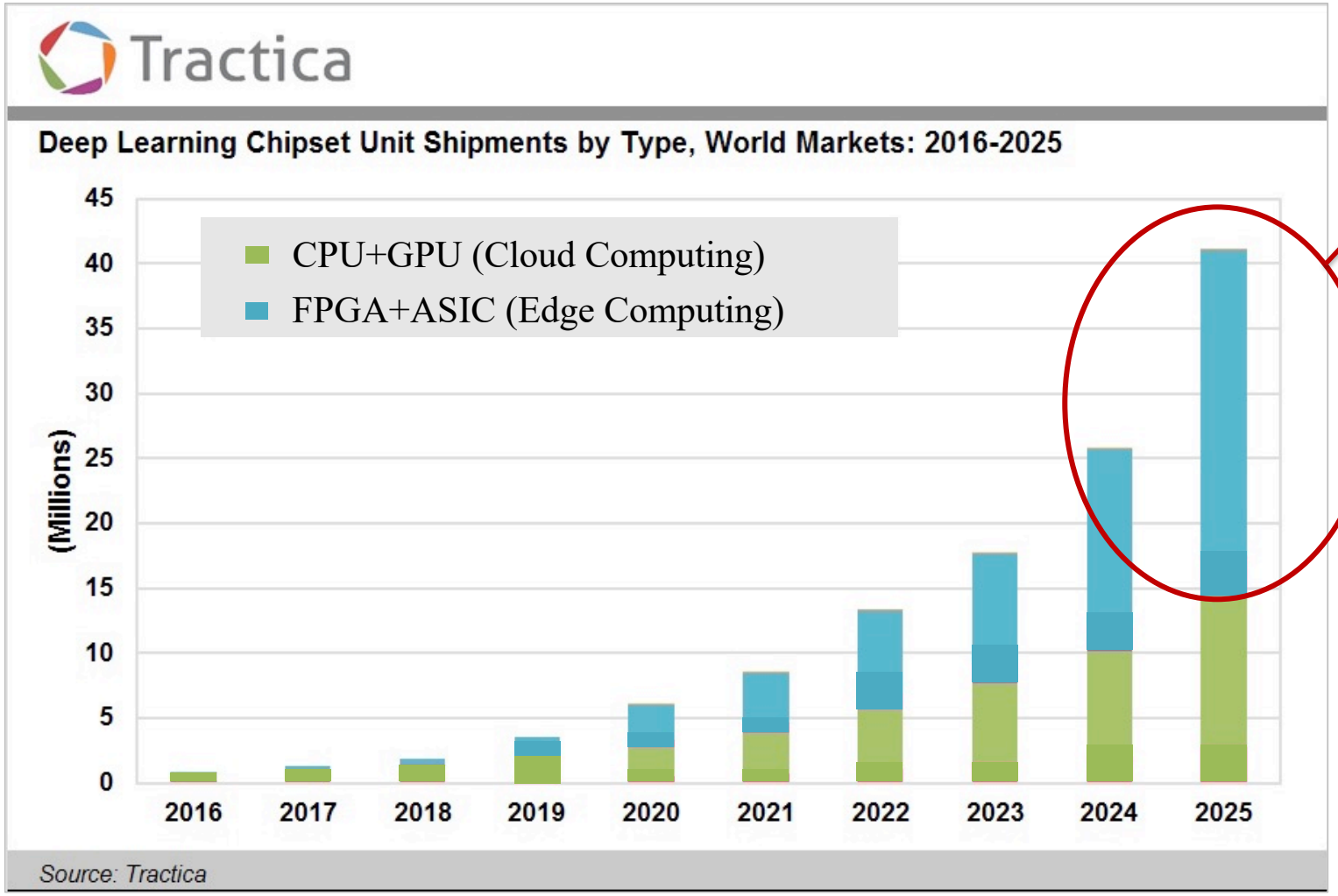
VISION: ML in ICD Devices

- **Accuracy -->** Higher than 72% (Cardiologist)
- **Area -->** Less than $3\mu m^2$
- **Power -->** Less than $20\mu W$
- **Latency -->** Less than 10 heartbeat

Voice language translation in AI Glasses

Surveillance Self-Driving Smart Home Crime Detection Beauty

Edge AI Will Have a Large Market



- By 2022, Edge AI will have the largest market
- Edge AI: Starting from 2018
  - Emerging field
  - Require a lot of research efforts

Opportunities on Edge AI

We Focus on Edge AI

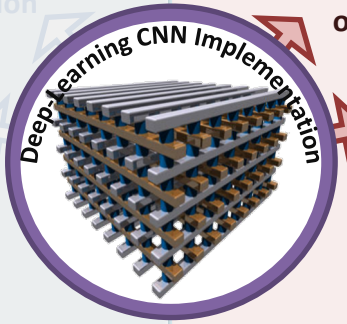
Larger Network and Hardware
for higher accuracy & lower latency

Tradeoff

Smaller Network and Hardware
for smaller area & lower power

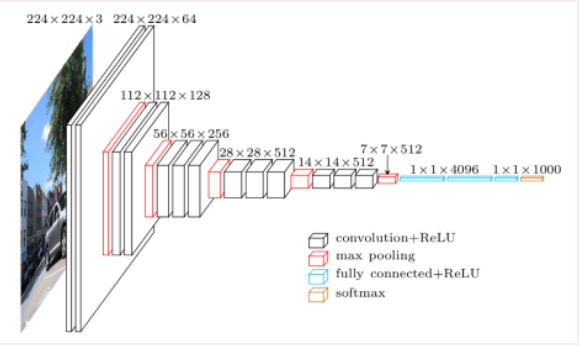
Brain-Inspired Computing

brain simulation



object detection

public safety and healthcare



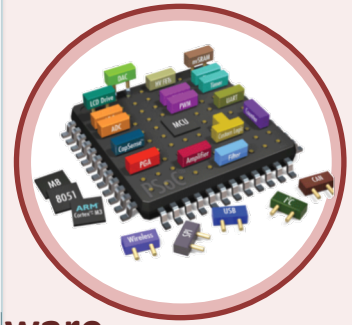
Demands:
Accuracy, Latency
Area, Power

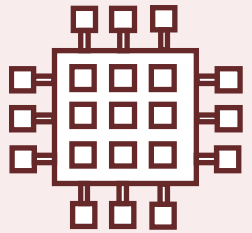
Edge Computing

CPU

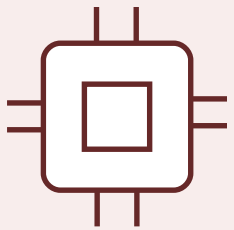
GPU

Hardware





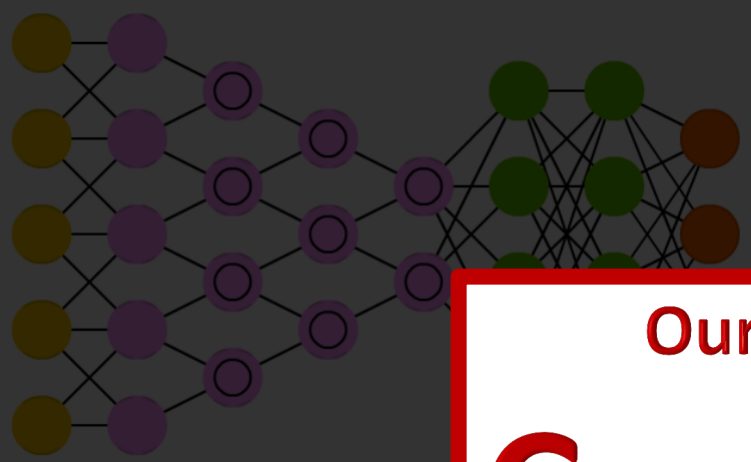
FPGA

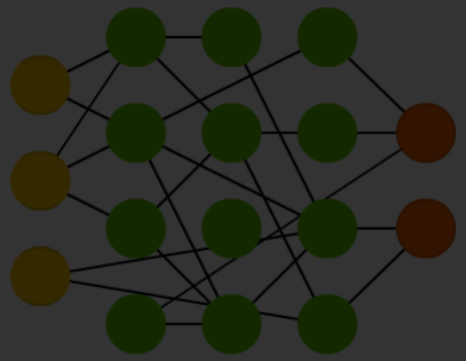


ASIC

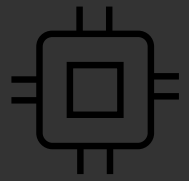
Tradeoffs Between Accuracy and Hardware Efficiency

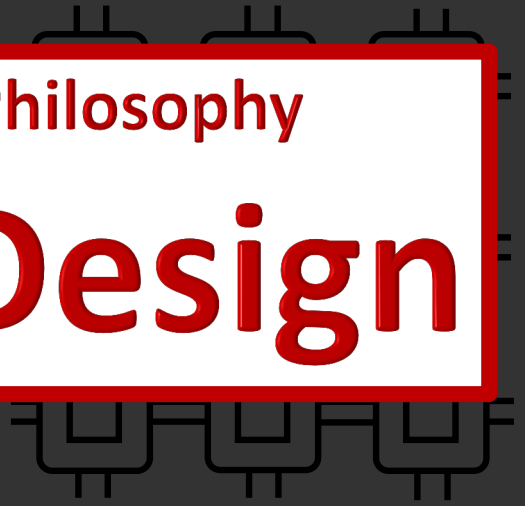
Neural Architectures

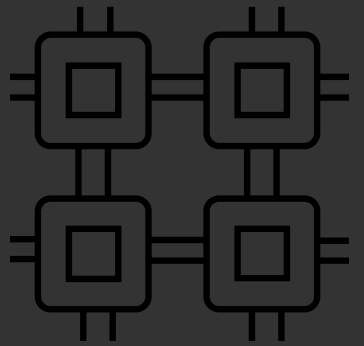




Hardware Design







Our Philosophy

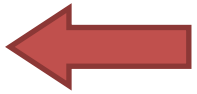
Co-Design

Metrics	Performance
Accuracy	++++ +
Area	++++ +
Power	++++ +
Latency	−−−− −

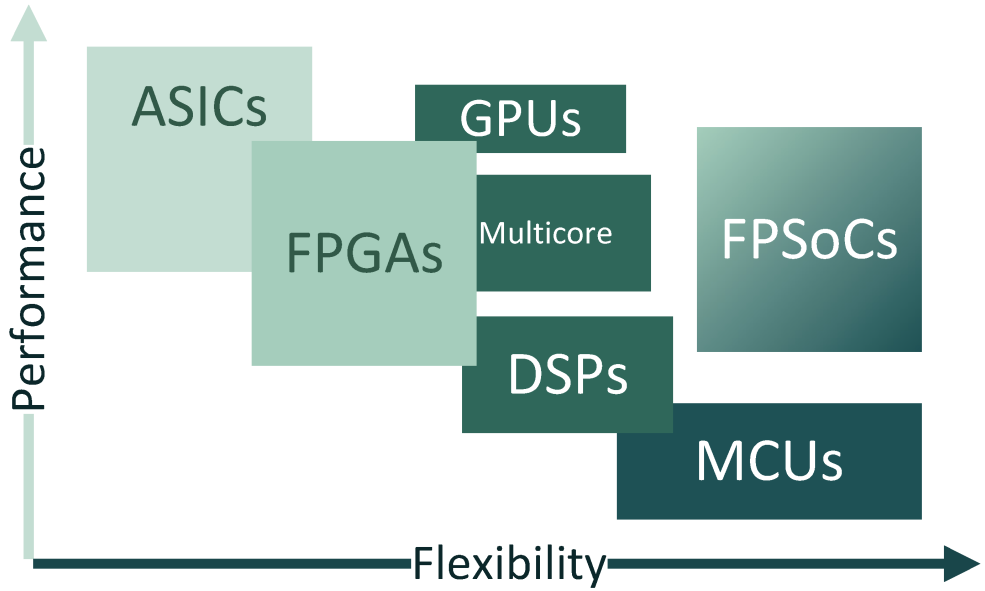
Metrics	Performance
Accuracy	−−−− −
Area	−−−− −
Power	−−−− −
Latency	++++ +

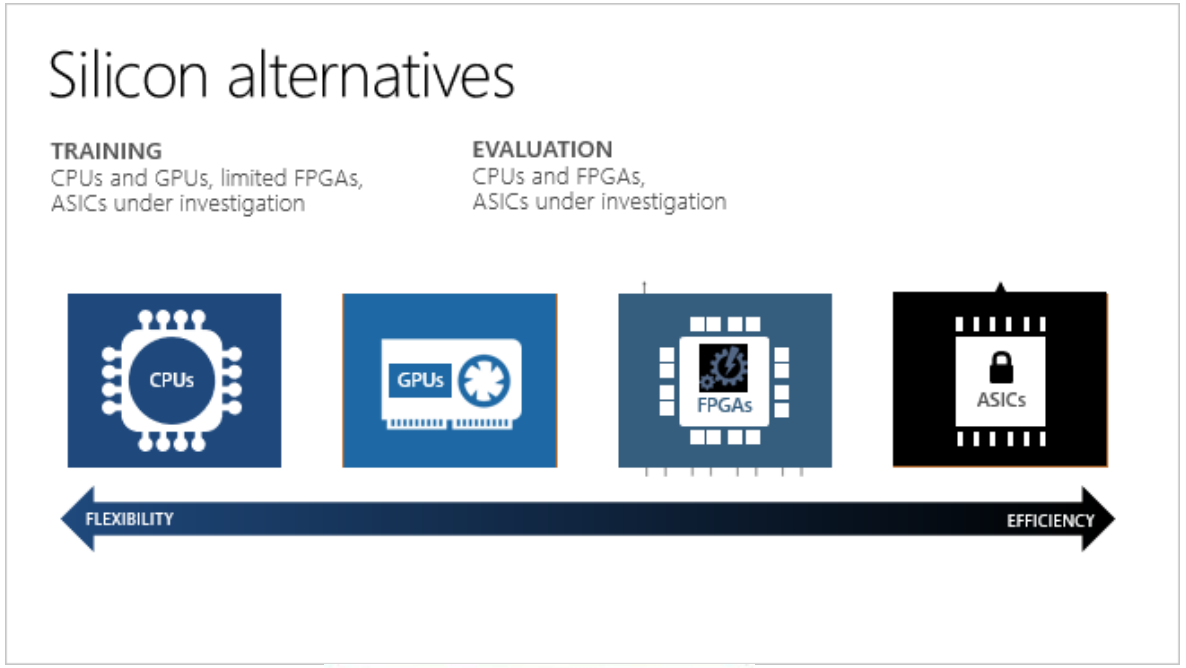
Metrics	Performance
Accuracy	++ +
Area	++ +
Power	++ +
Latency	++ +

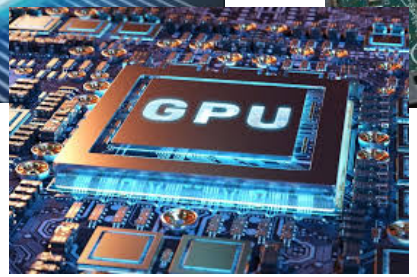
OUTLINE

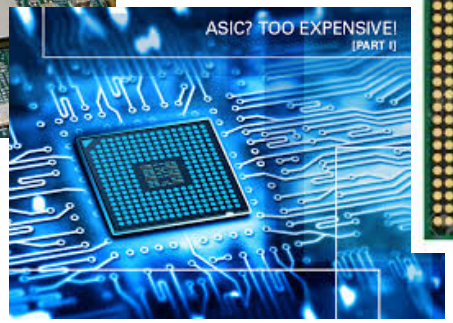
- Hardware Design Space 
- Software Design Space
- First Step: Co-Exploration of Neural Architecture and FPGAs
- Second Step: Co-Exploration of Neural Architecture and Heterogenous ASICs
- Third Step: Co-Exploration of Neural Architecture and Scalable ASICs
- Future Work

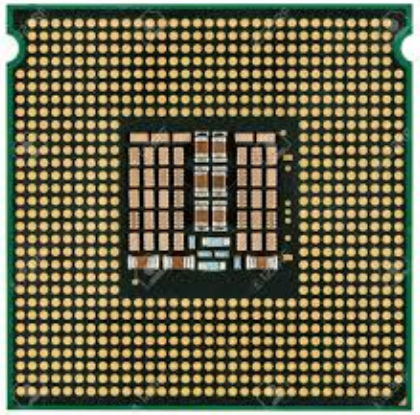
Complex Design Space with kinds of Computing systems





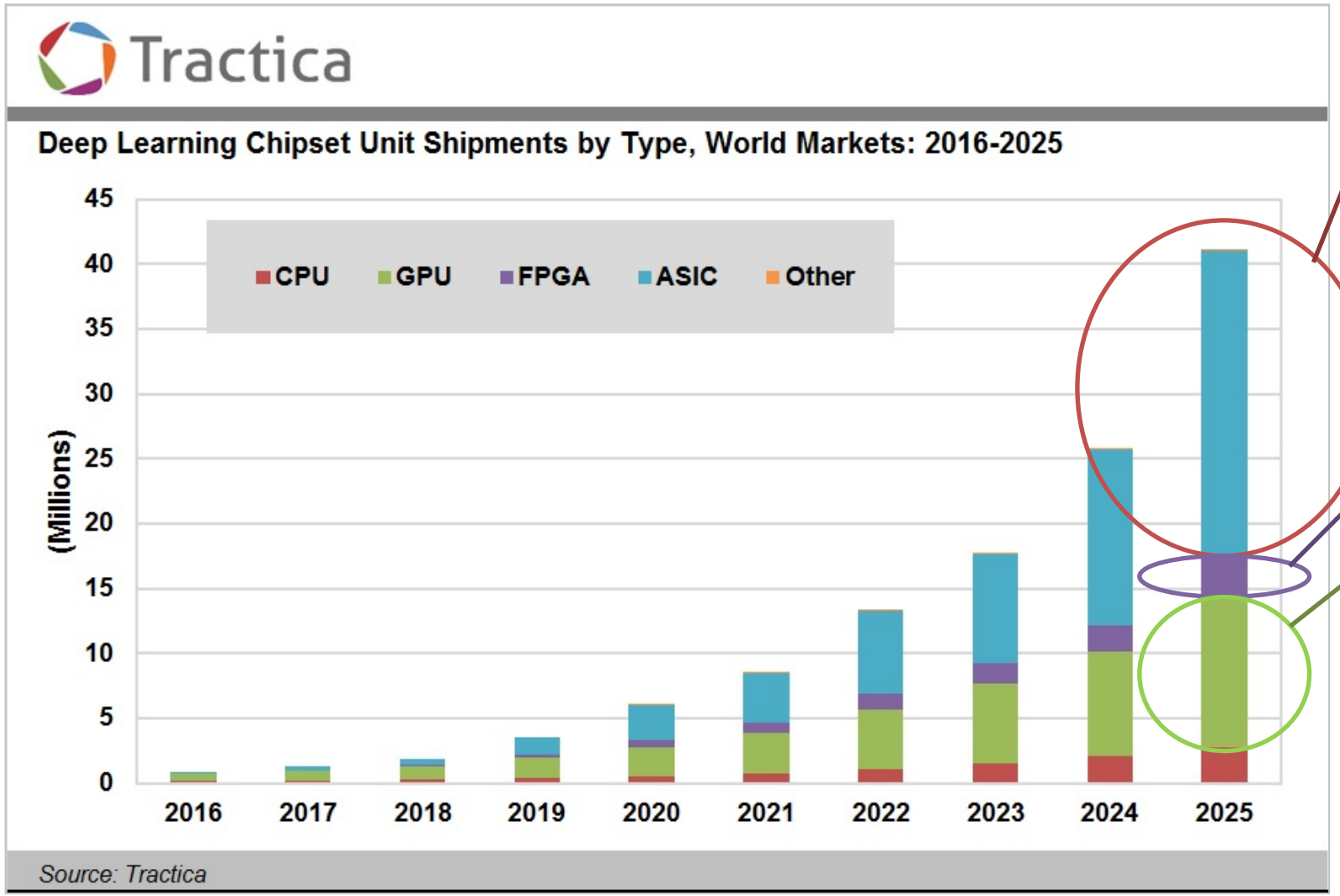






.....

ASIC Will Have the Largest Market



ASIC

FPGA

GPU

- **ASICs have larger market than FPGA, GPU, CPU for AI applications**
  - High Energy Efficiency
  - Low Latency
  - Small Area

Design Space of FPGAs, ASICs and multicores is

Huge

Dataflow (data reuse):

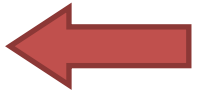
- Weight stationary
- Output stationary
- Row stationary
- No reuse

y:

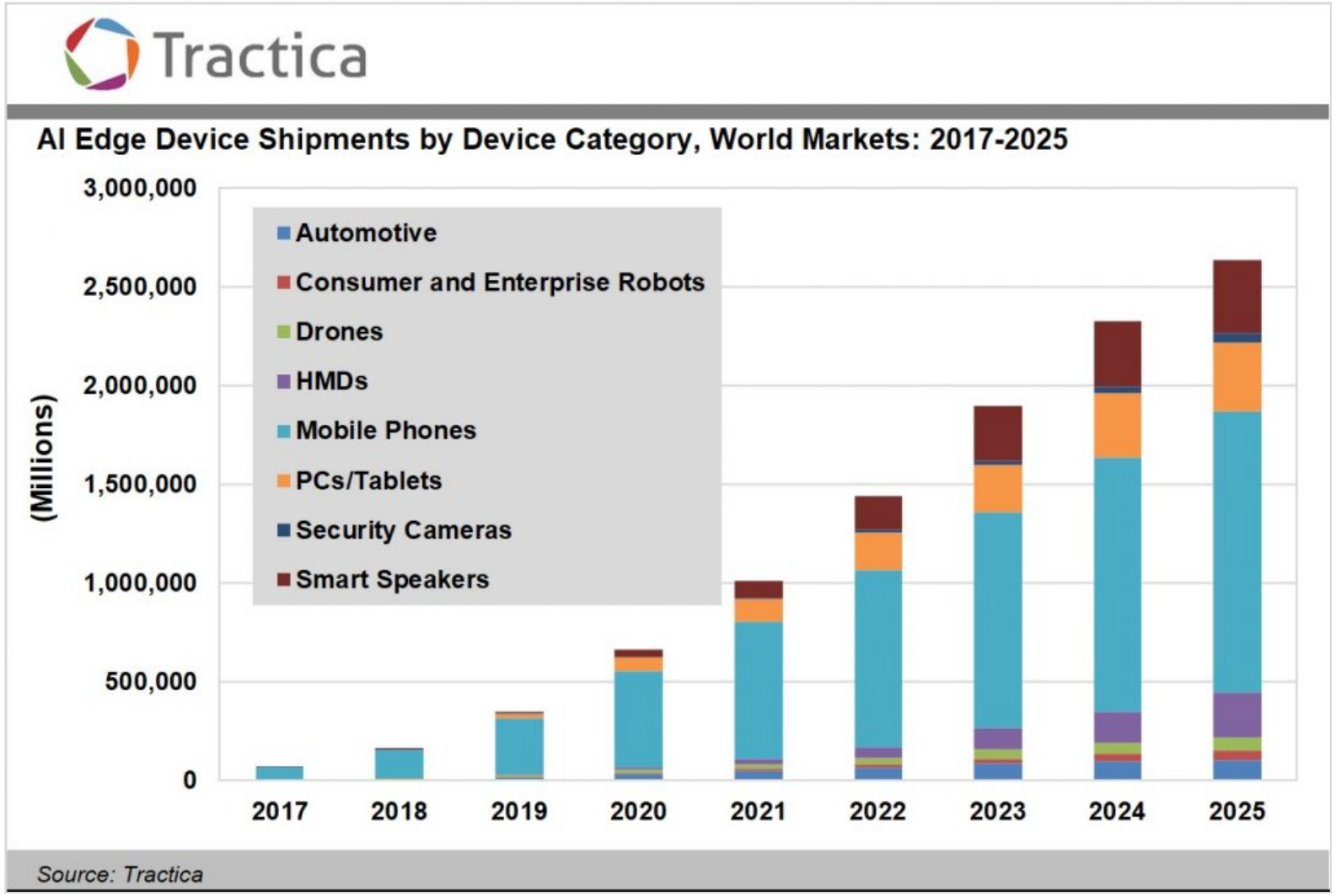
Communication:

- Wormhole
- Store and forward

OUTLINE

- Hardware Design Space
- Software Design Space 
- First Step: Co-Exploration of Neural Architecture and FPGAs
- Second Step: Co-Exploration of Neural Architecture and Heterogenous ASICs
- Third Step: Co-Exploration of Neural Architecture and Scalable ASICs
- Future Work

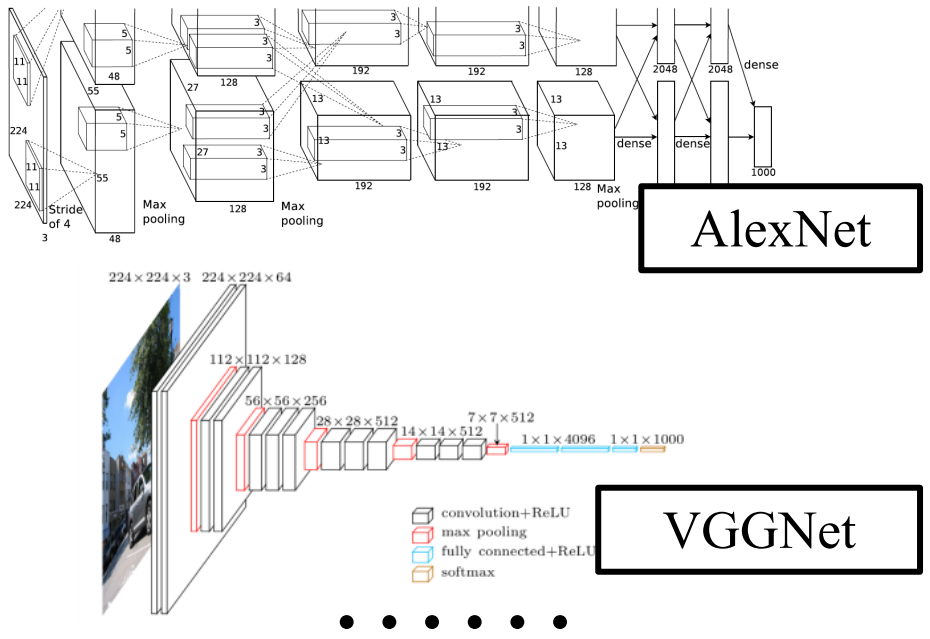
Dedicated Neural Networks for Categories of Applications



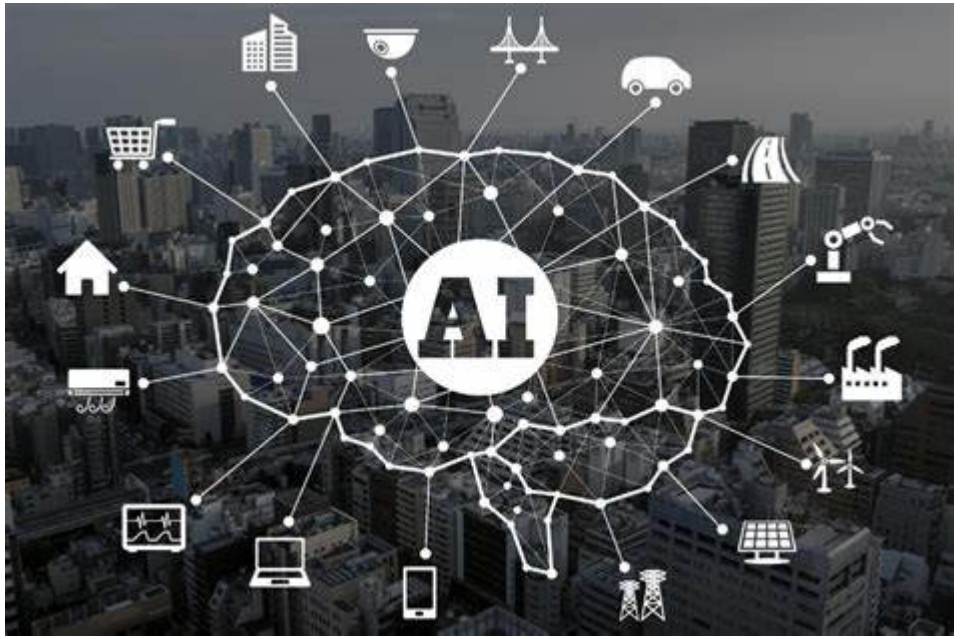
- Increased Categories of edge AI applications
- Applications need **dedicated neural networks** to adapt input data (e.g., Image, Voice)

- Design dedicated neural networks for specific apps?
- To achieve expected performance?

Human Invented Neural Architectures



Design arch. for dedicated application

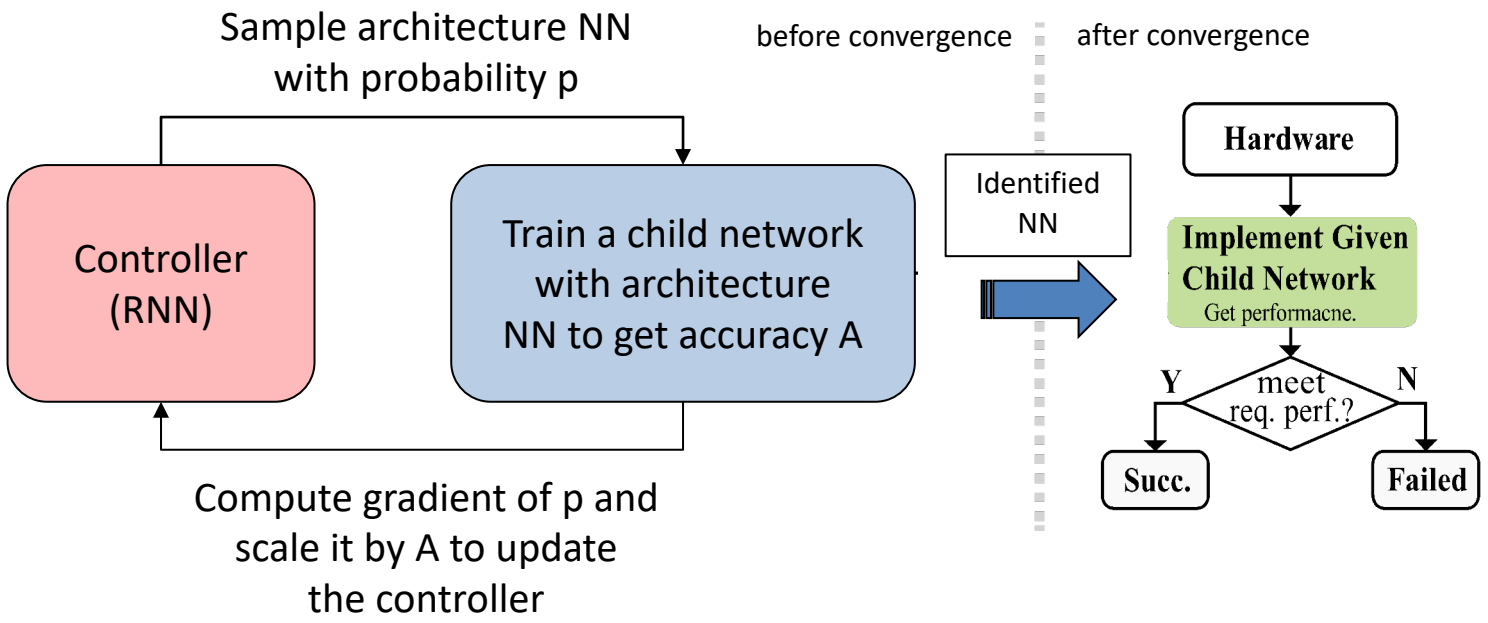


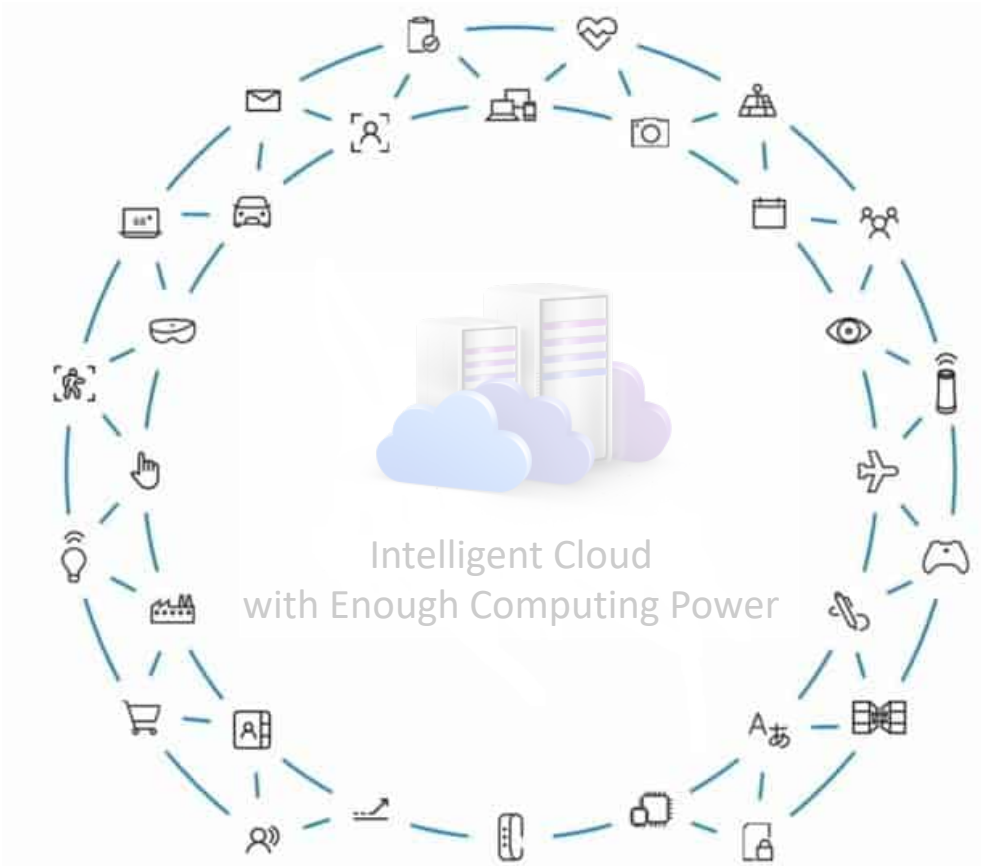
Era of AI Democratization

Problem

- **Domain knowledge and excessive labor**
- **It is impossible to manually design specific arch. for each dedicated application in the era of AI democratization**

Neural Architecture Search (NAS)





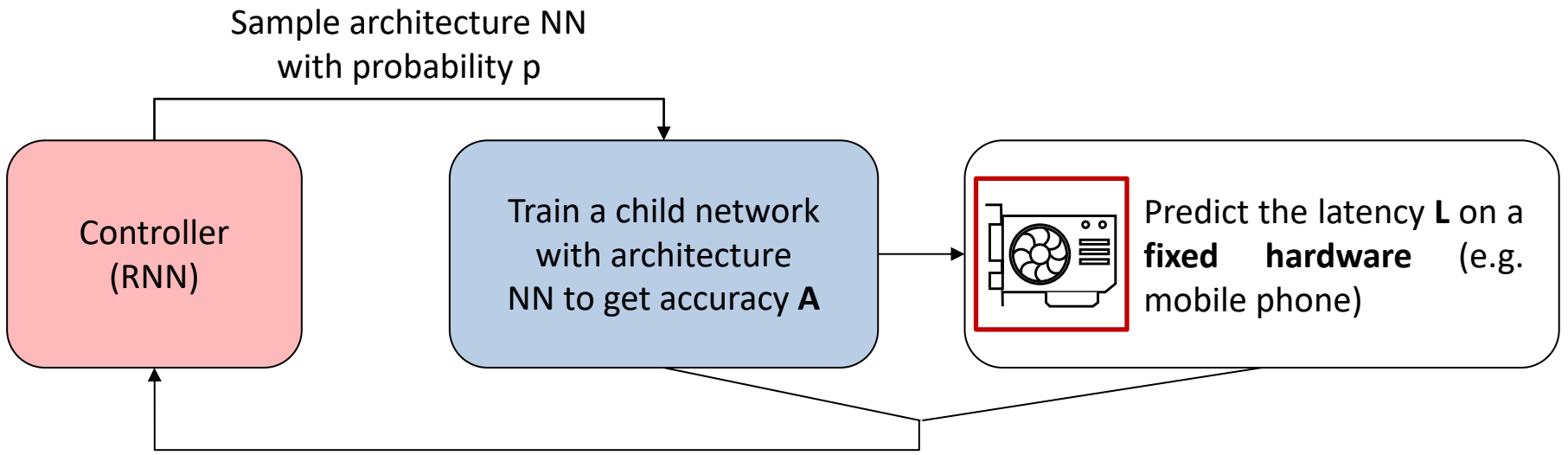
Intelligent Edge
with Limited Computing Power

Problem

- **No constraint on hardware resource consumed**

REF: Zoph, Barret, and Quoc V. Le. "Neural architecture search with reinforcement learning." *ICLR 2017*

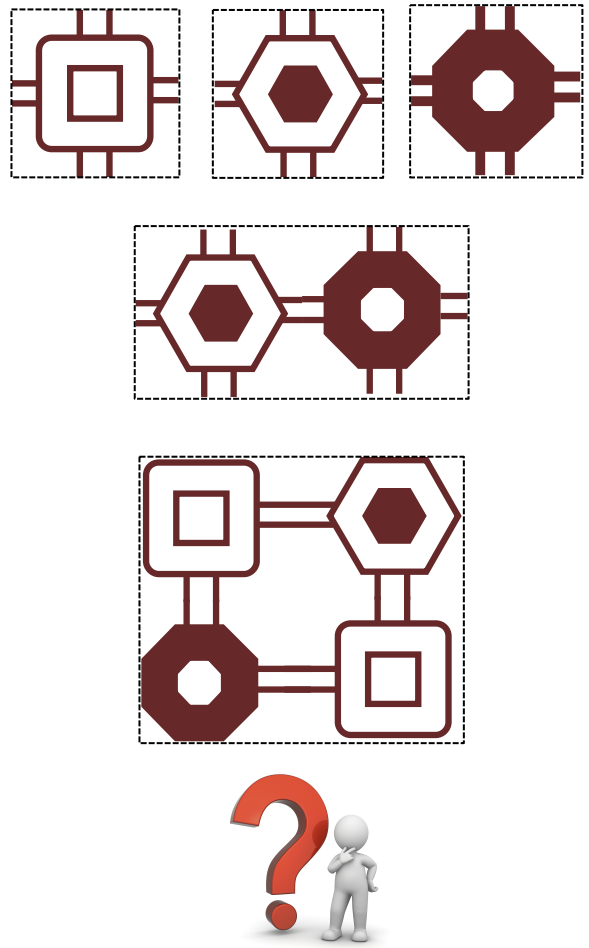
HARDWARE-AWARE NAS



Problem

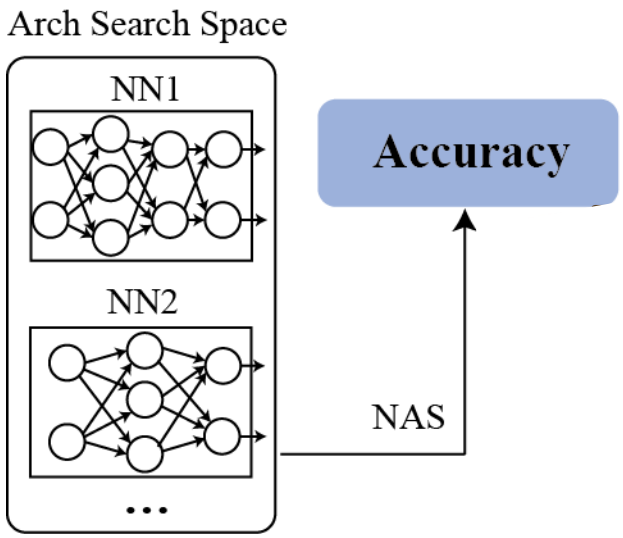
- It works for particular fixed hardware (e.g., CPU, GPU), but not suitable for designable hardware (e.g., FPGA, ASIC)

Different Hardware Designs (e.g., FPGA, ASIC)

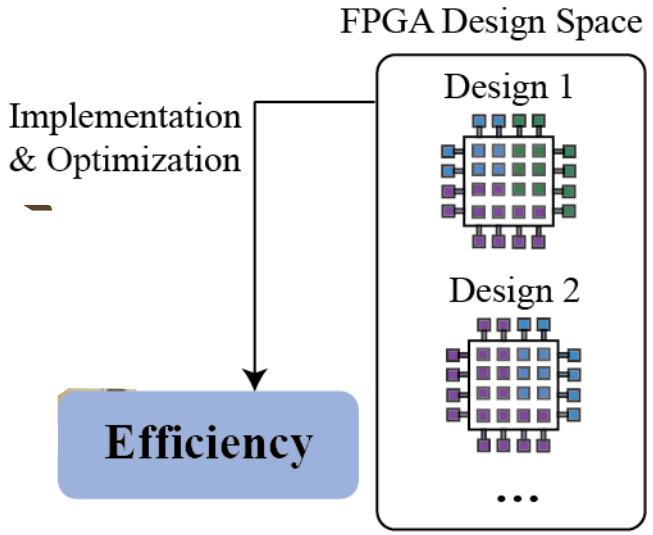


REF: Mingxing Tan, et al. "MnasNet: Platform-Aware Neural Architecture Search for Mobile." arXiv 2018

A Missing Link between Two Design Spaces

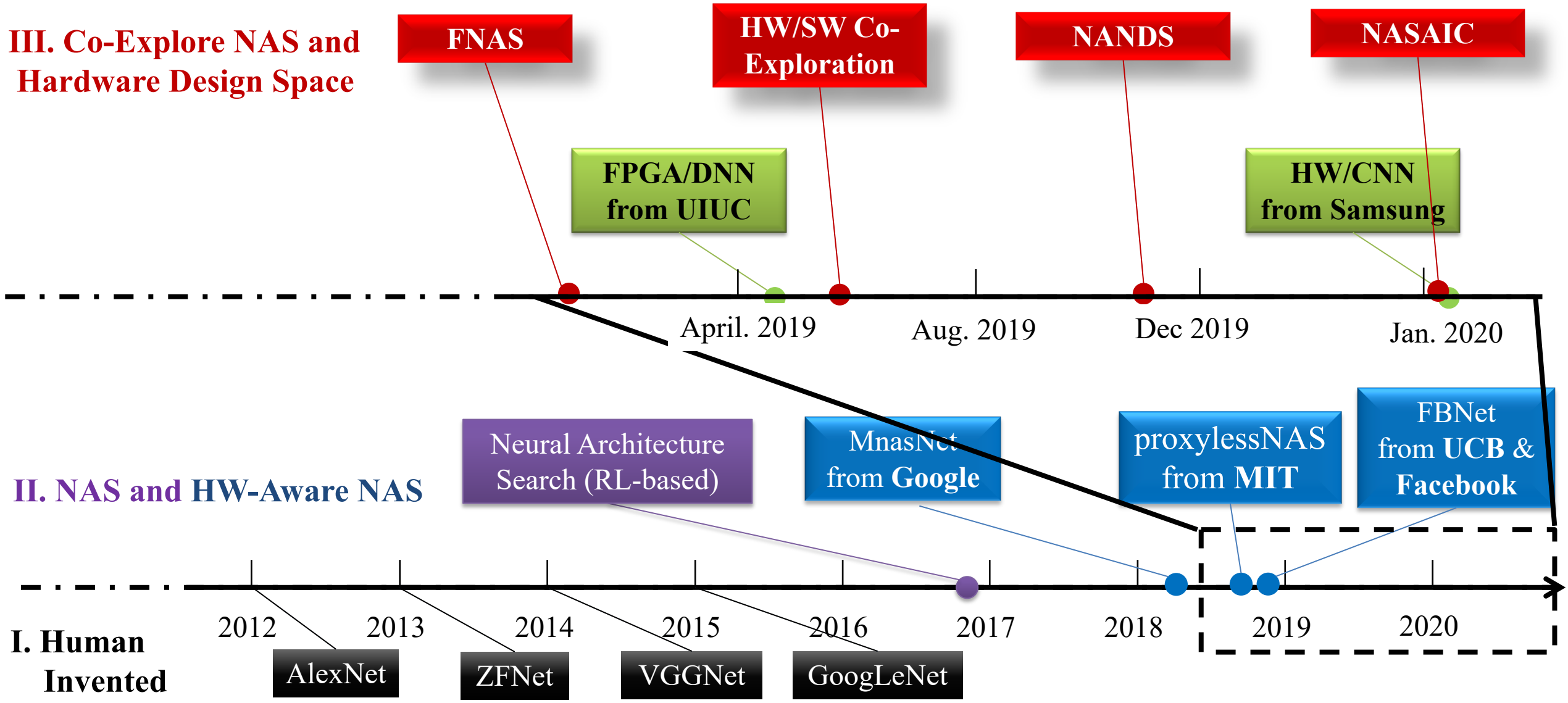


Neural Architecture Search

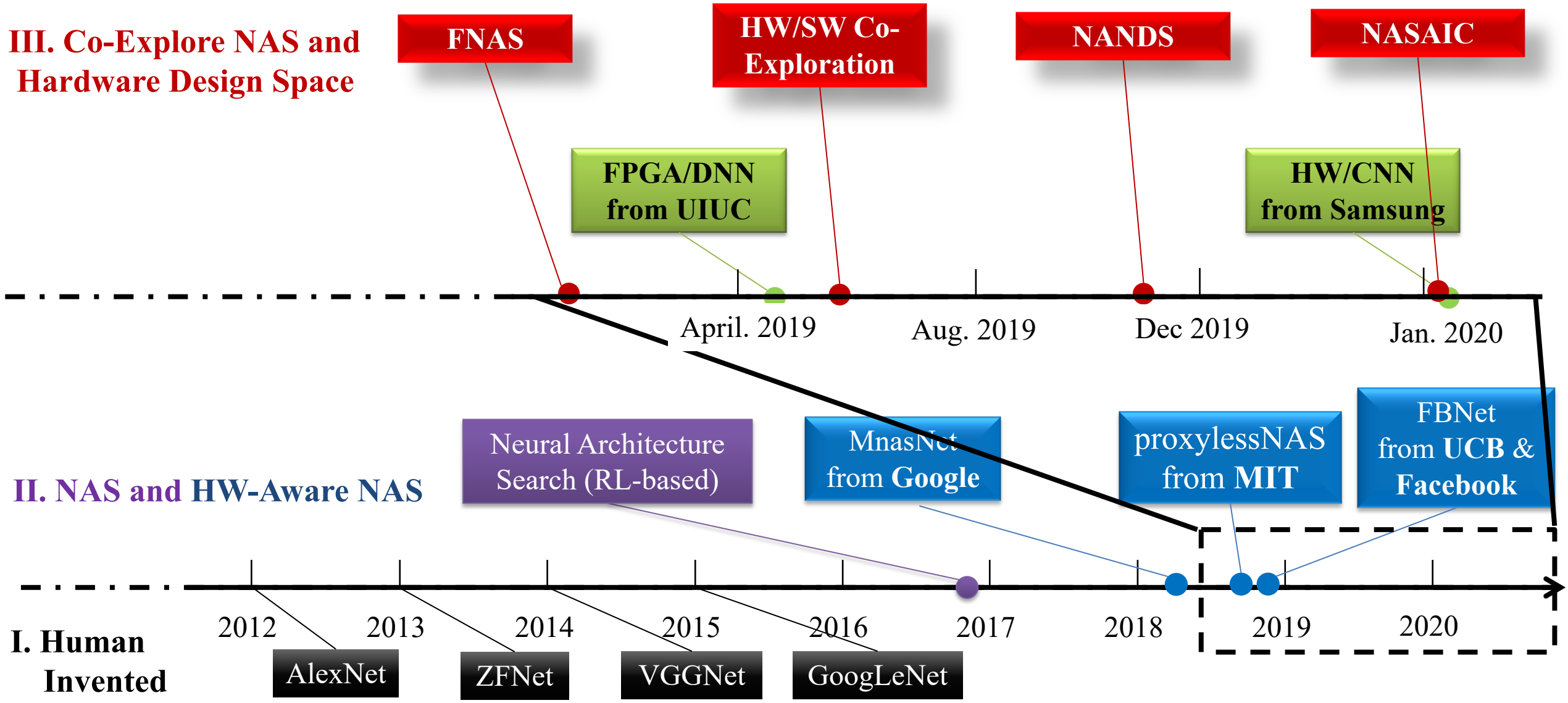


Neural Architecture Implementation on Hardware

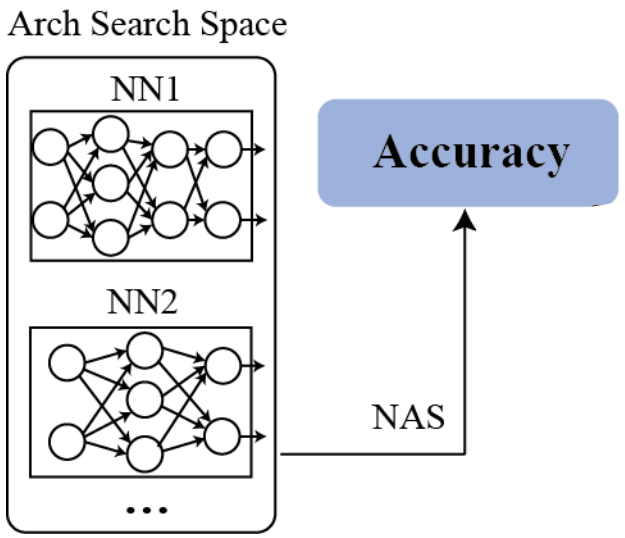
Evolution of Deep Neural Architecture Exploration



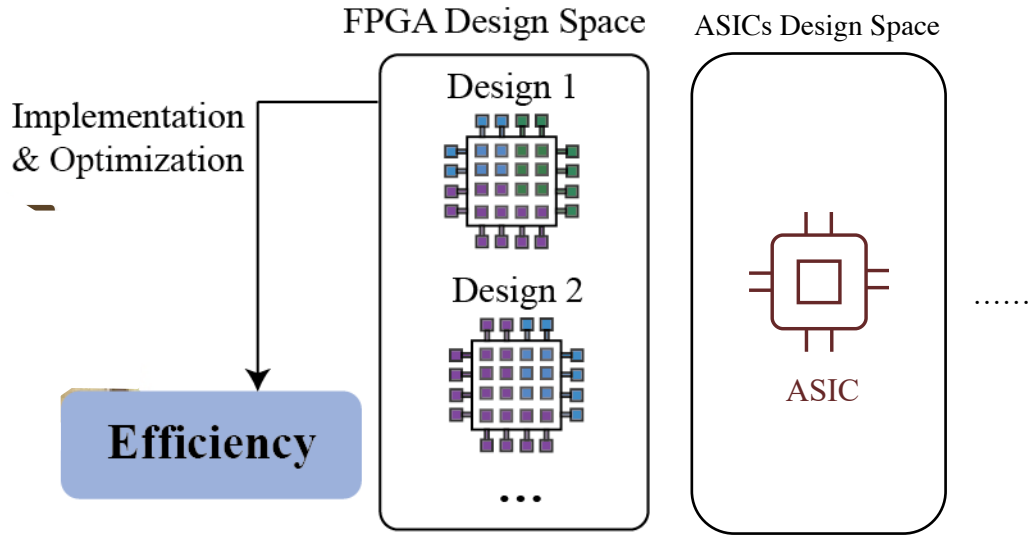
Evolution of Deep Neural Architecture Exploration



A Missing Link between Two Design Spaces

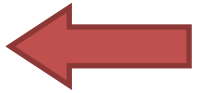


Neural Architecture Search



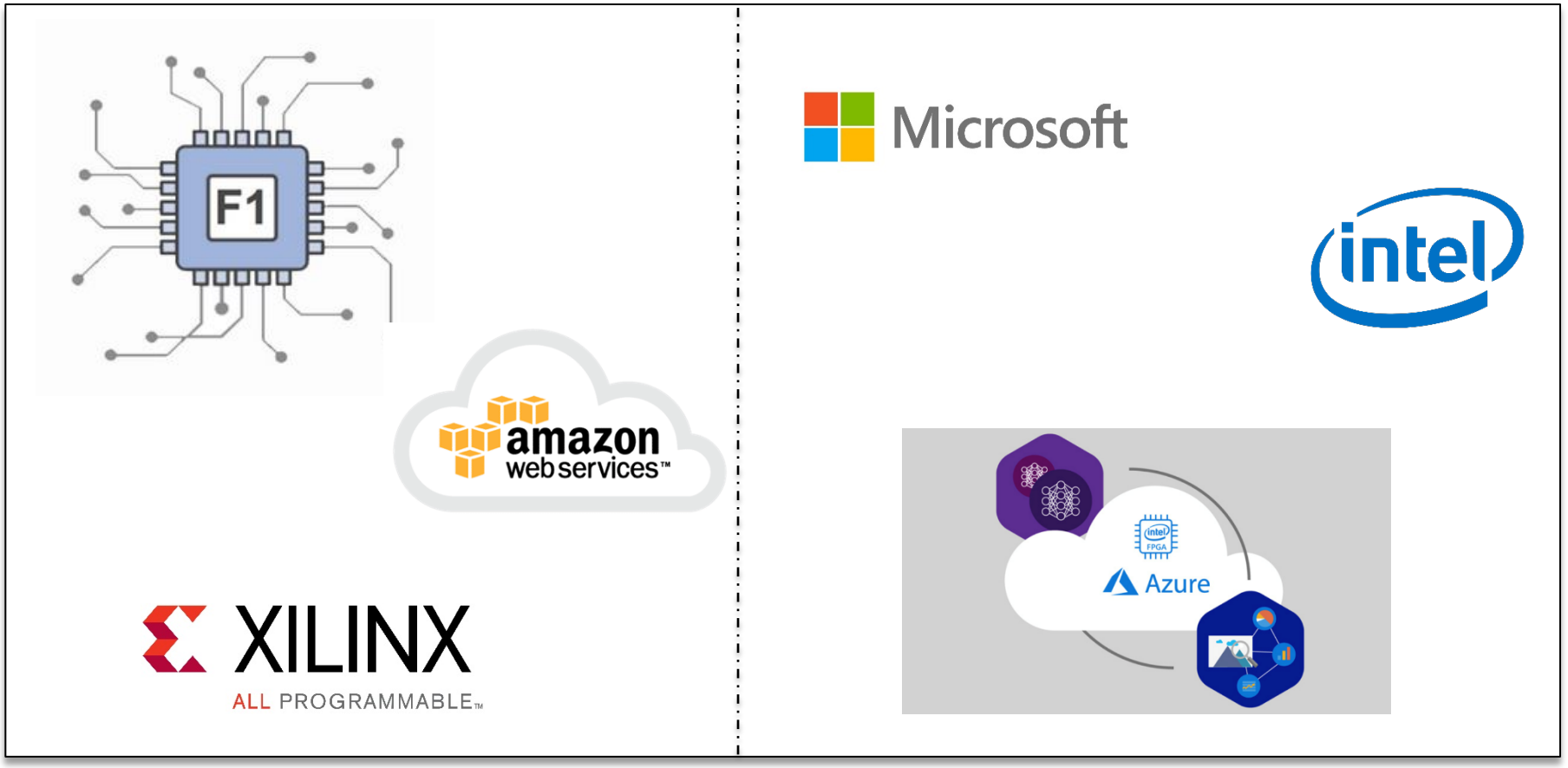
Neural Architecture Implementation on Hardware

OUTLINE

- Hardware Design Space
- Software Design Space
- First Step: Co-Exploration of Neural Architecture and FPGAs 
- Second Step: Co-Exploration of Neural Architecture and Heterogenous ASICs
- Third Step: Co-Exploration of Neural Architecture and Scalable ASICs
- Future Work

FPGAs in DNN Applications

FPGA in Cloud Computing



[ref] Where Do FPGAs Stand in Auto IC Race? https://www.eetimes.com/document.asp?doc_id=1333419#
[ref] PYNQ in UAV. http://brennancain.com/pynqcopter-an-open-source-fpga-overlay-for-uavs/

FPGAs in DNN Applications

FPGA in Cloud Computing

F1

amazon web services™

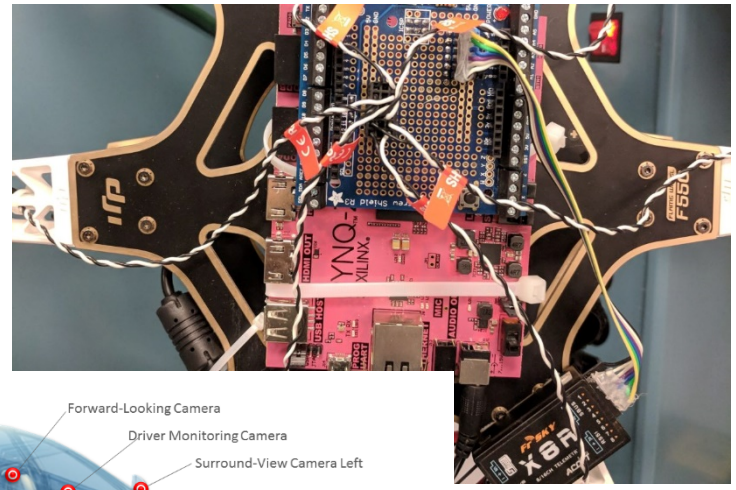
Microsoft

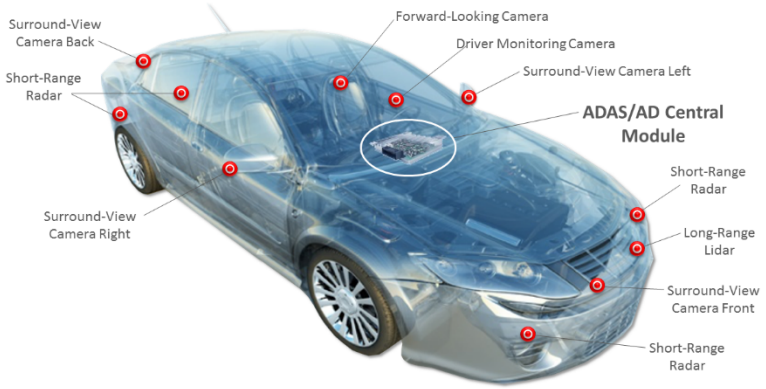
intel

Azure

XILINX
ALL PROGRAMMABLE™

FPGA in Edge Computing

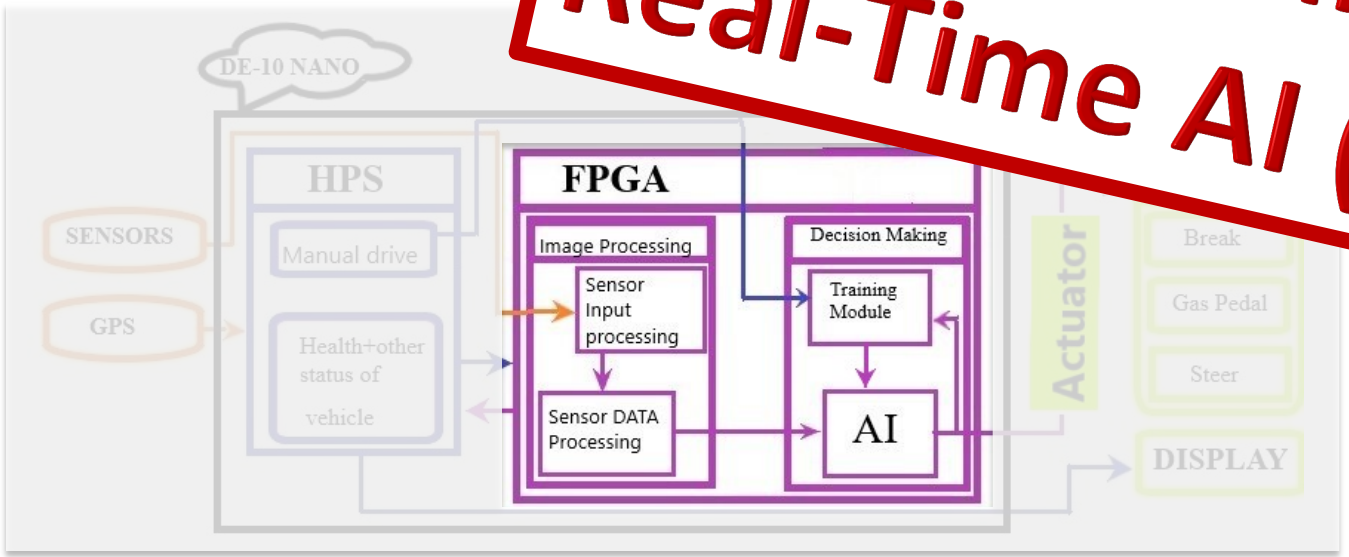




[ref] Where Do FPGAs Stand in Auto IC Race? https://www.eetimes.com/document.asp?doc_id=1333419#
[ref] PYNQ in UAV. http://brennancain.com/pynqcopter-an-open-source-fpga-overlay-for-uavs/

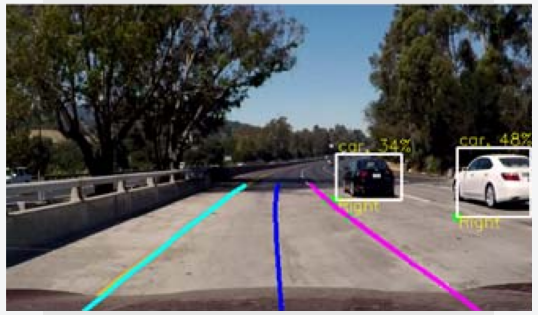
FPGAs in Edge AI: Autonomous Car

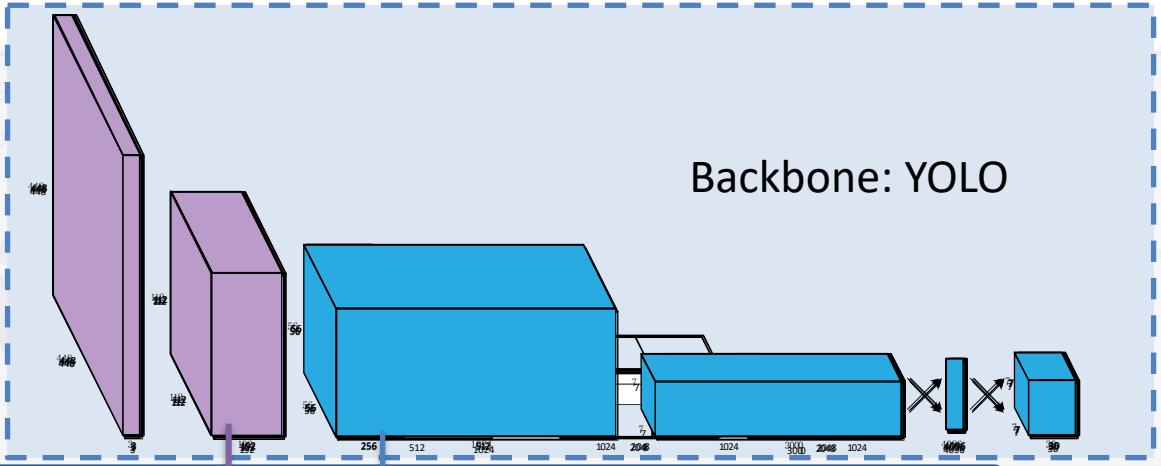
New Challenge: Real-Time AI (Latency)



Source: www.innovatefpga.com

HW/SW Co-Exploration

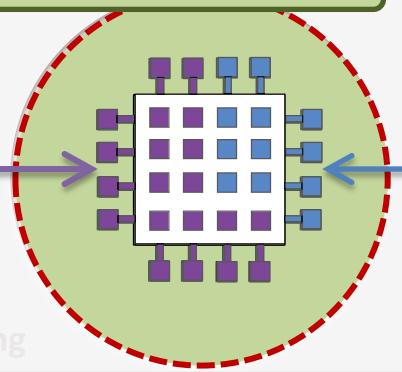




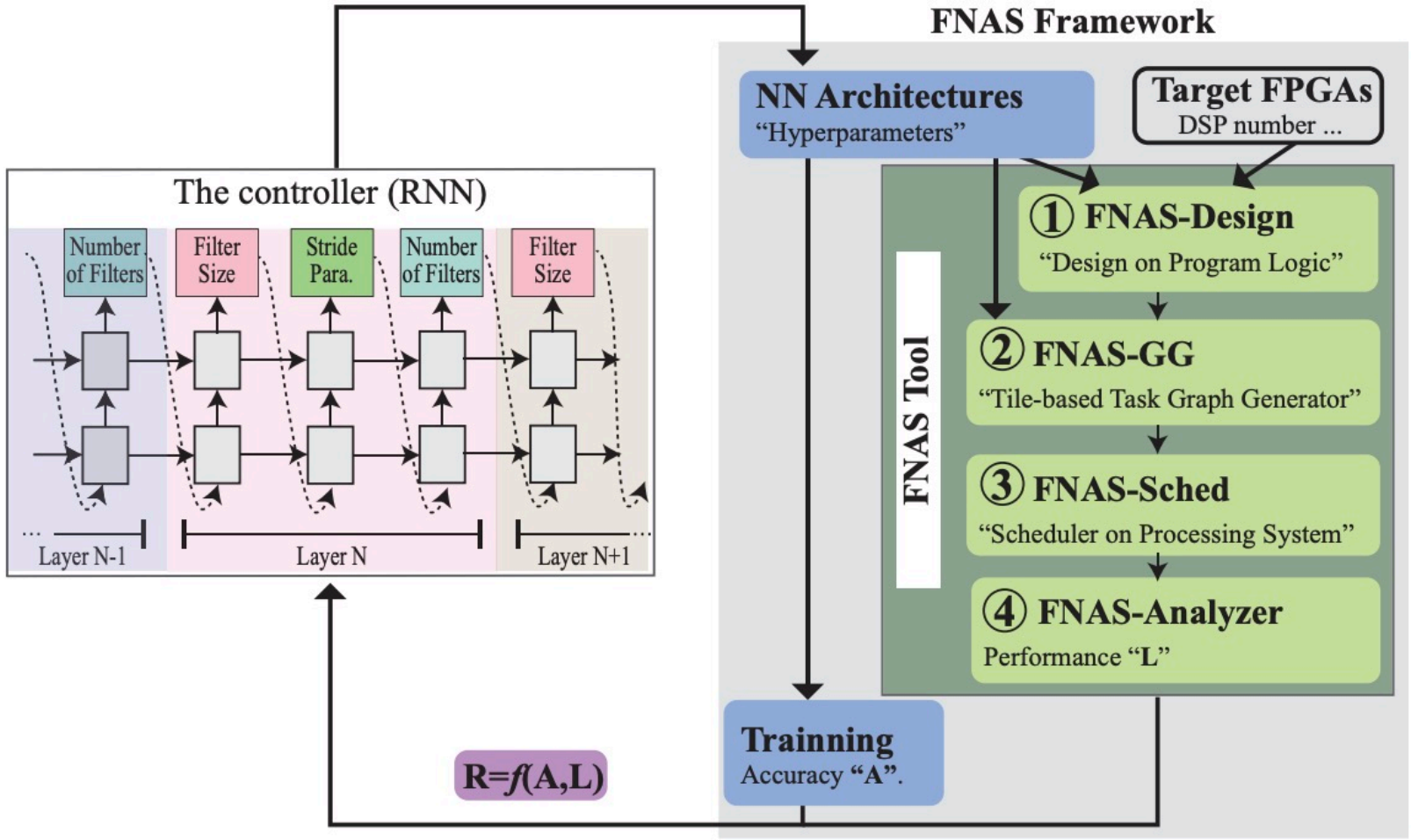
Neural Architecture Search Space

Co-Exploration

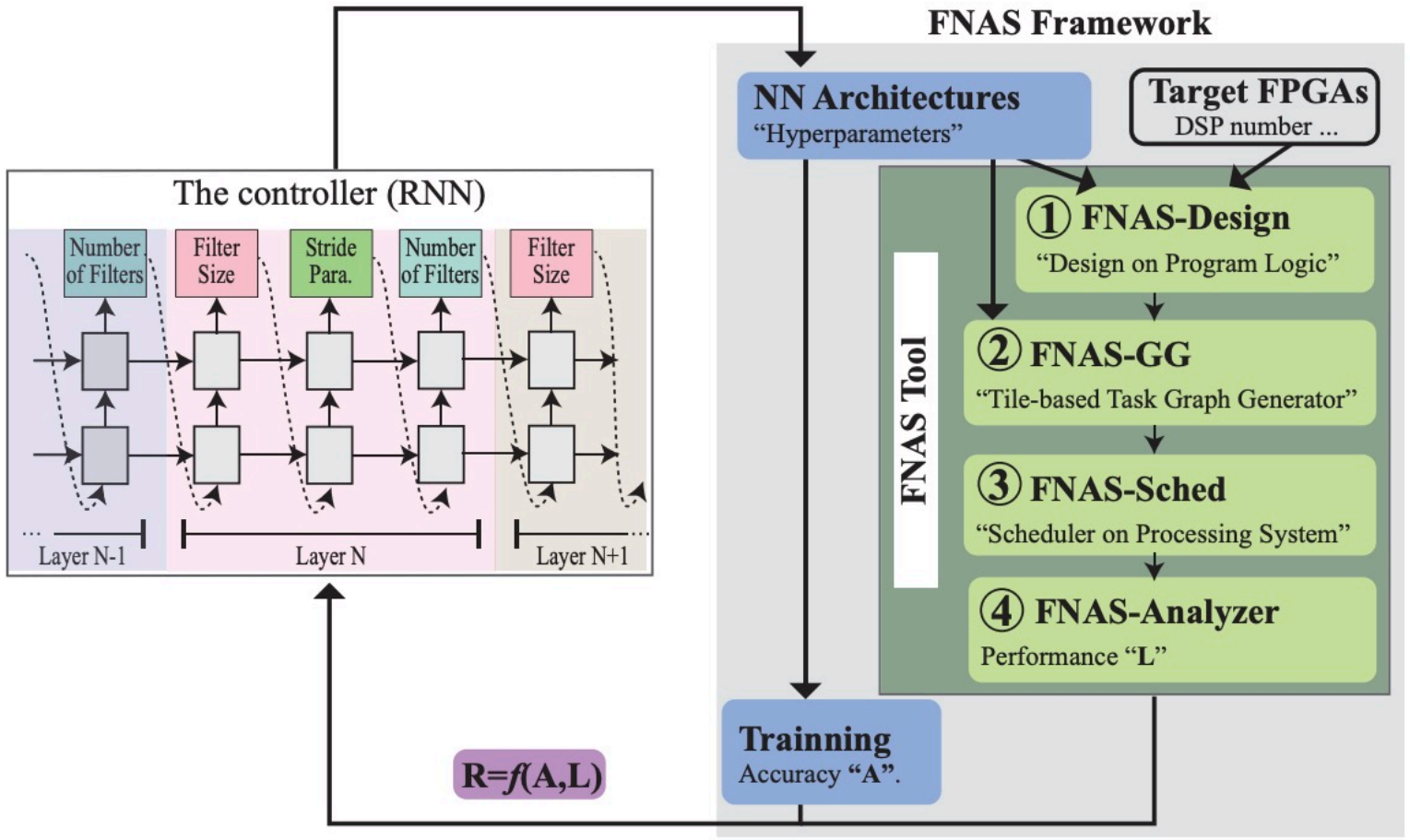
Hardware Design Space



Co-Exploration Framework



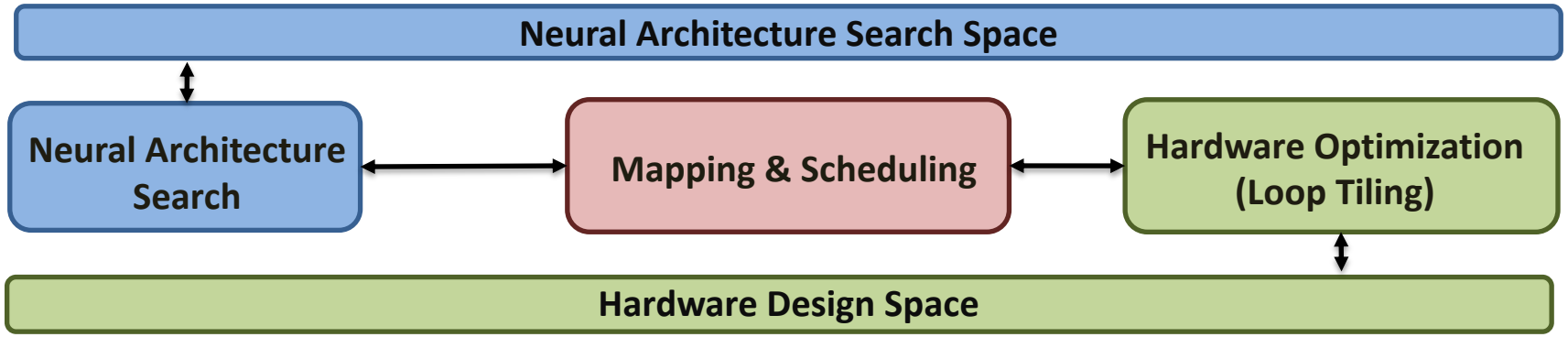
- CO-EXPLORATION FRAMEWORK

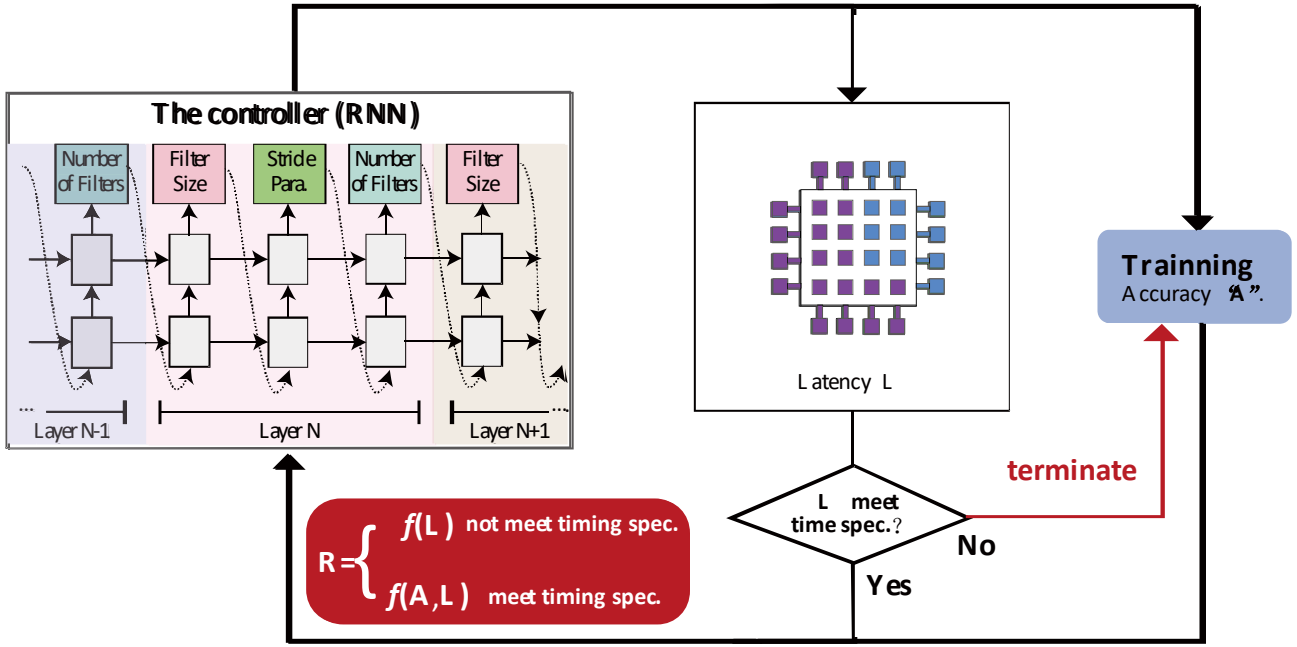


Reward Function:

$$R = \begin{cases} \frac{rL-L}{rL} - 1 & L > rL \\ (A-b) + \frac{L}{rL} & L \leq rL \end{cases}$$

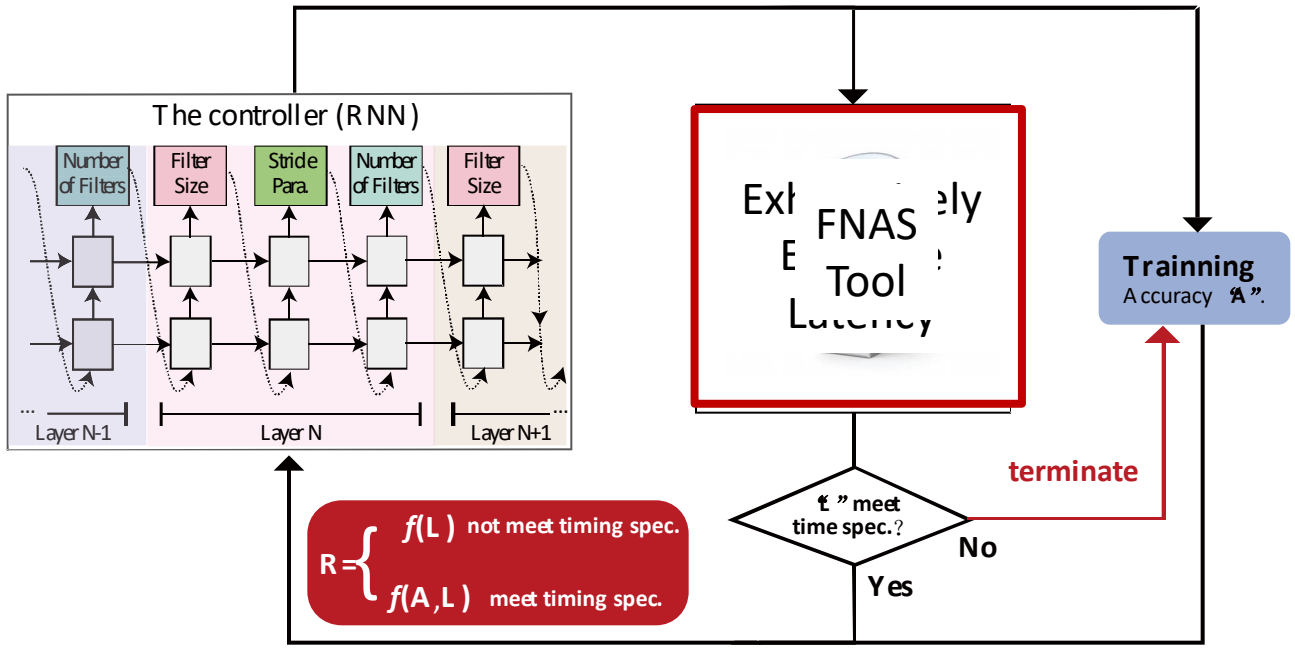
Co-Exploration Framework





- **Apply Reinforcement Learning based NAS.**
  - Can be any other search method, like evolutional algorithms, differentiable architecture search.
- **Add Hardware Design Module.**
  - Can terminate lengthy training if latency cannot satisfy real-time constraints.
- **Modify Reward Function.**
  - Latency and Accuracy can be simultaneously optimized.

PERFORMANCE ESTIMATION: SOLUTIONS & CHALLENGES



Our Solution: FNAS tools to response to challenges

FNAS-Design C1
"Design on Program Logic"

FNAS-GG C2
"Tile-based Task Graph Generator"

FNAS-Sched C2
"Scheduler on Processing System"

FNAS-Analyzer C3
Estimate Performance "L"

Naïve Solution: HW-Aware + Exhaustively Evaluate Lat.

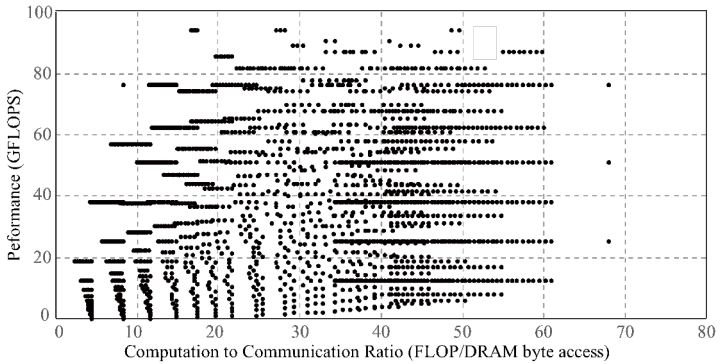


Fig1. Possible designs for Layer 5 of AlexNet on ZCU102

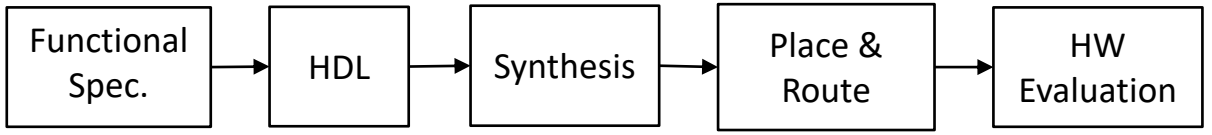


Fig2. Procedure of performance evaluation

Challenges:

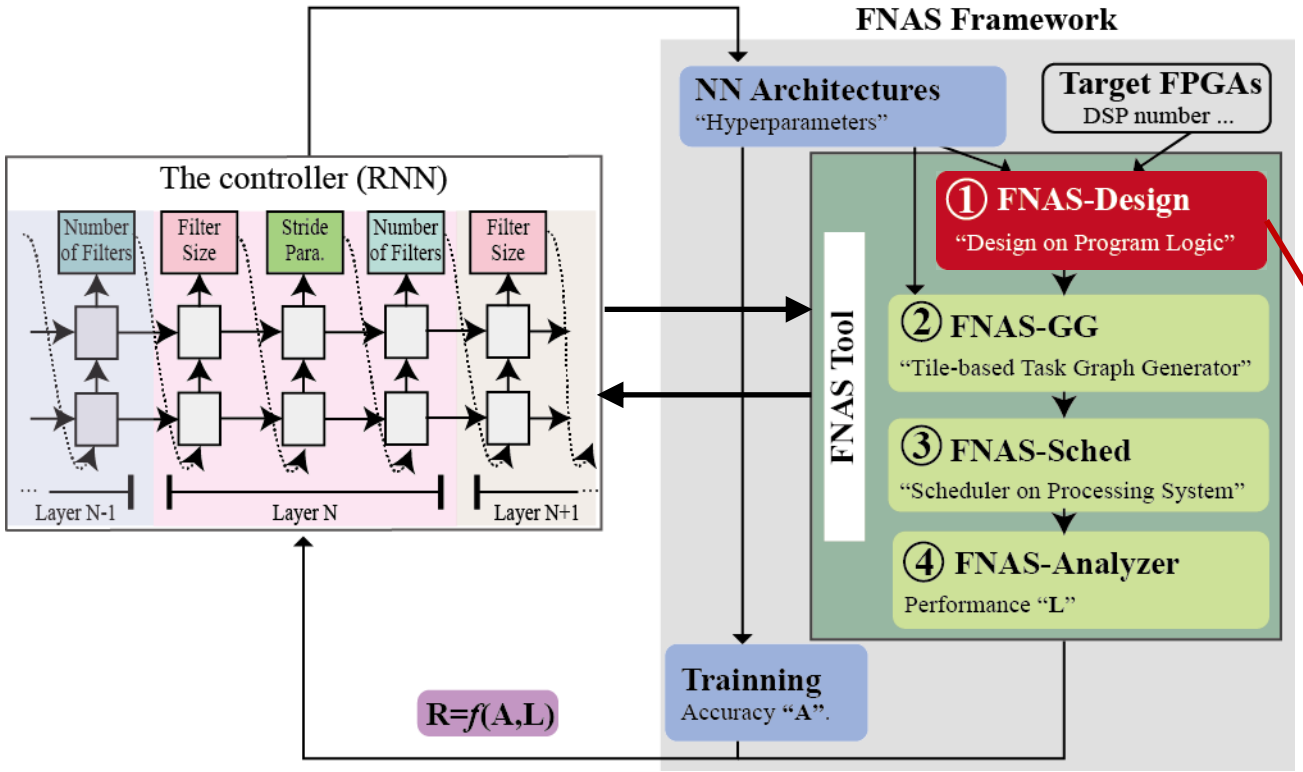
C1: Huge design space!

C2: Multi-FPGA design!

C3: Time-consuming evaluation!

Infeasible

FNAS: Design Optimization



① FNAS Design

Designed to determine the tiling parameters for a given NN architecture on target FPGAs.

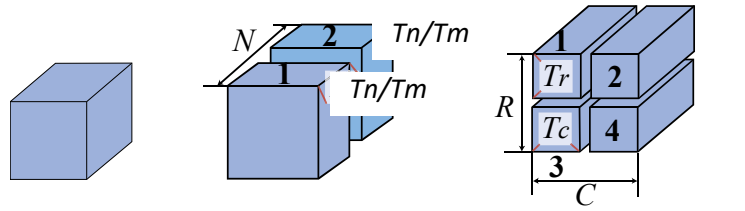
On-chip accelerator design:

Determine：

1. On-chip buffer allocation; 2. Accelerator size for computing

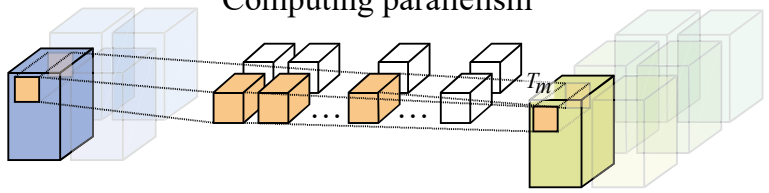
(note: both are determined by tiling parameters, Tm, Tn, Tr, Tc)

One layer:



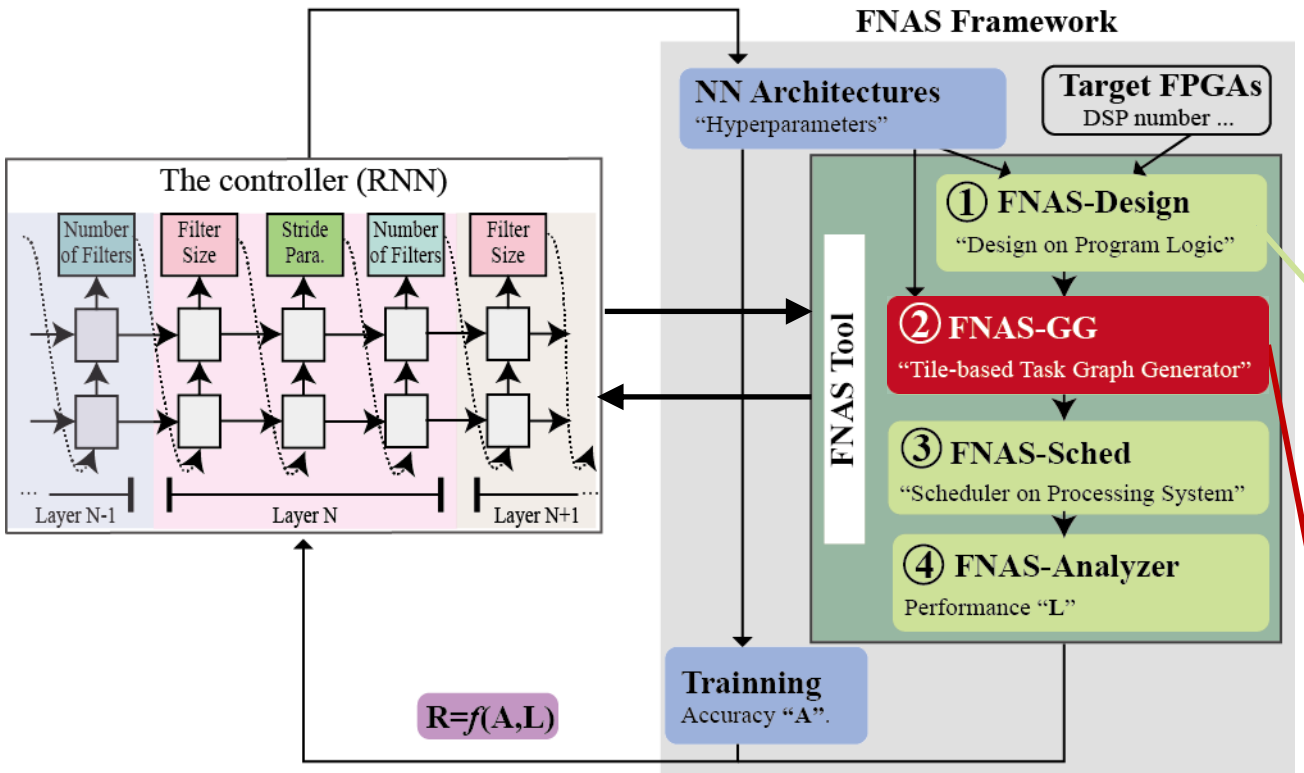
Computing parallelism

Multiple layers:



REF: Chen Zhang et al. 2015. Optimizing fpga-based accelerator design for deep convolutional neural networks. In Proc. of FPGA.

FNAS: Graph Generator

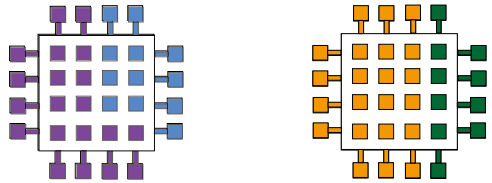


② FNAS GG

a graph generator that takes the design parameters and NN architecture to generate the dependency graph between data tiles and tasks

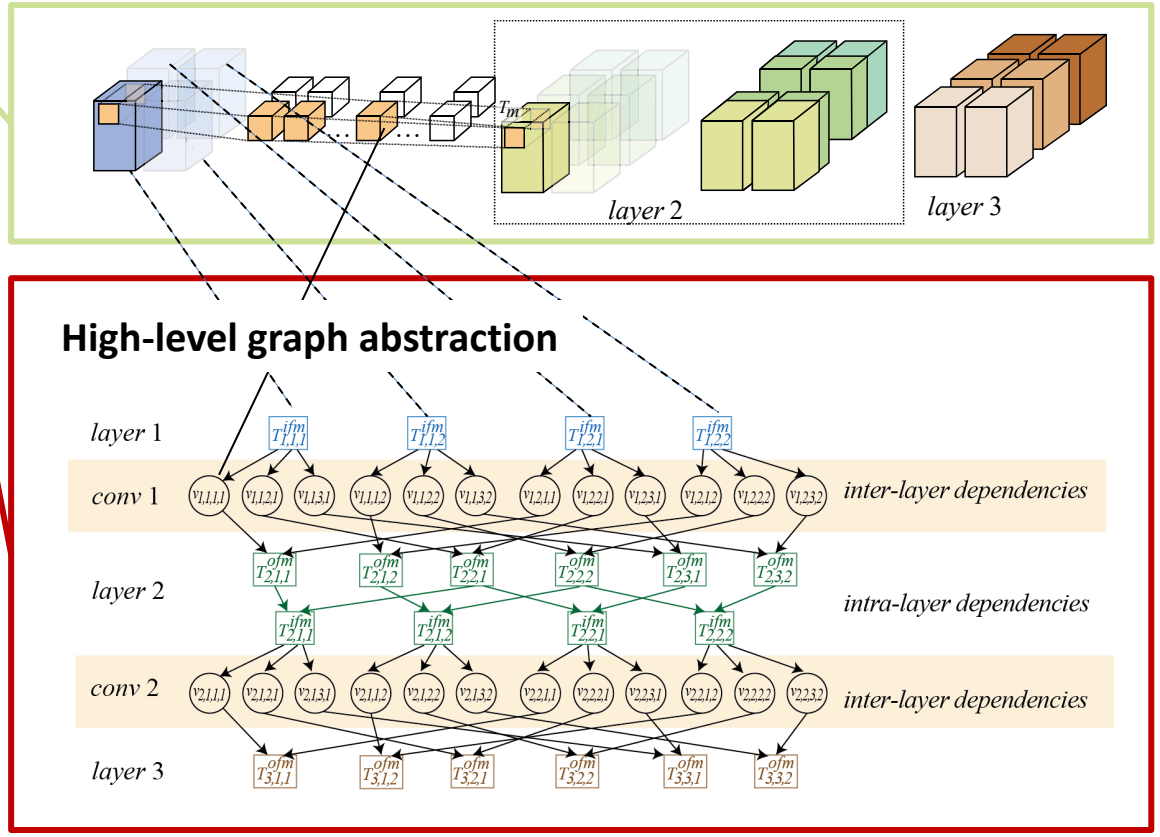
Given:

1. FPGAs with attributes including LUTs, DSPs, BRAM, etc.

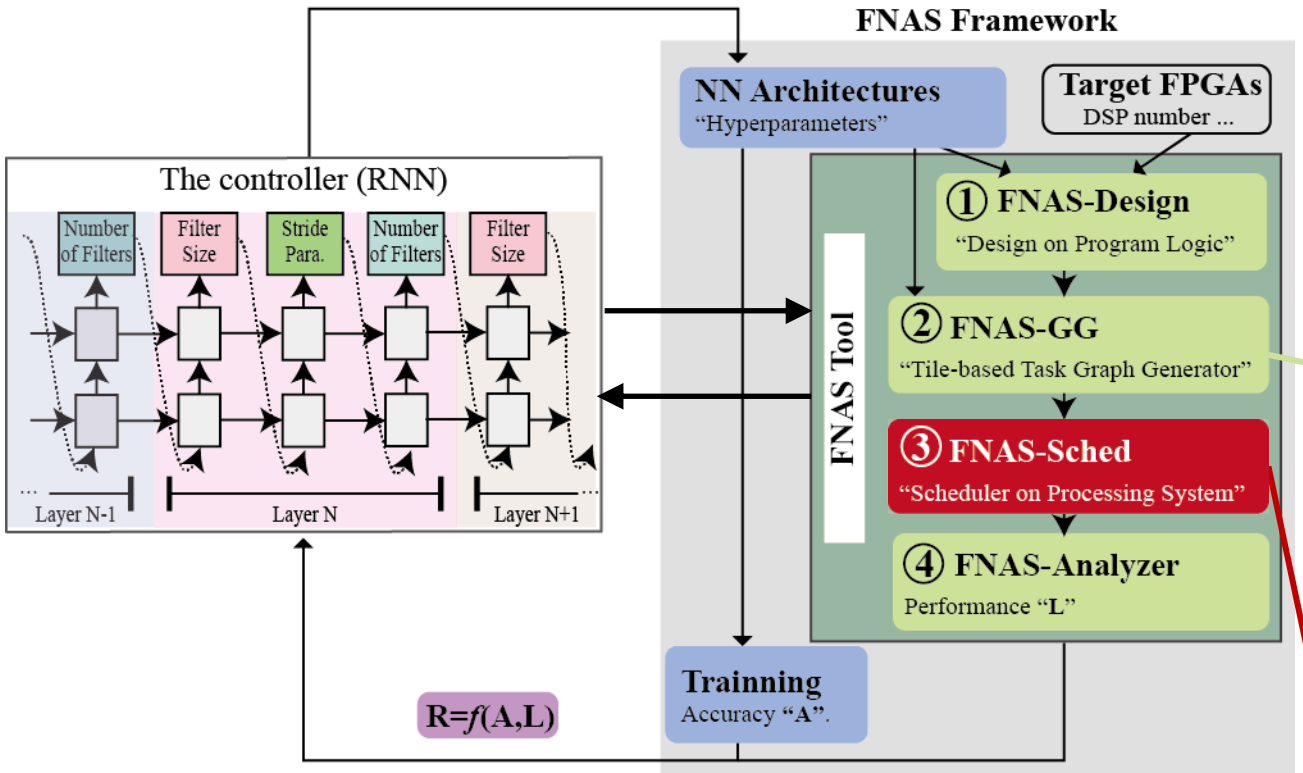


.....

2. A neural architecture with determined hyperparameters

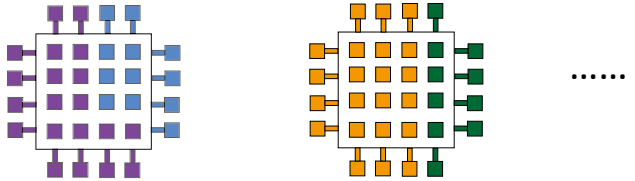


FNAS: SCHEDULE

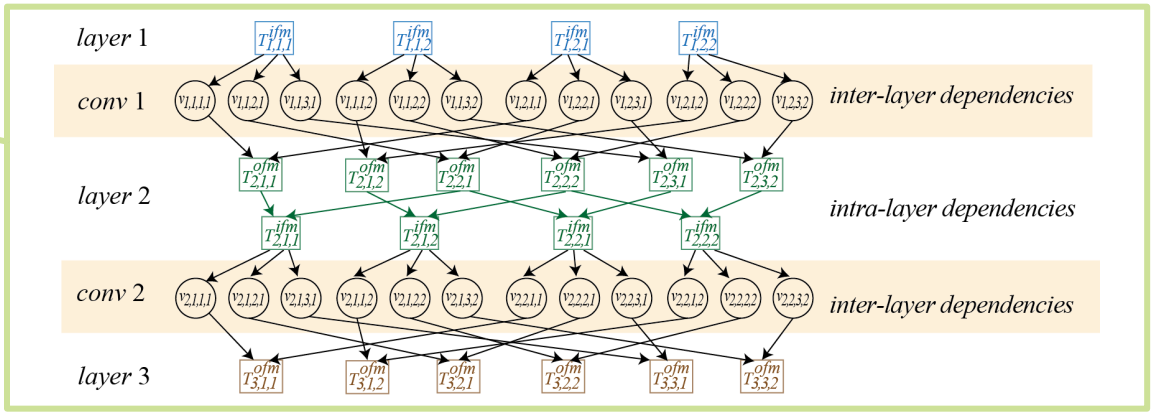


Given :

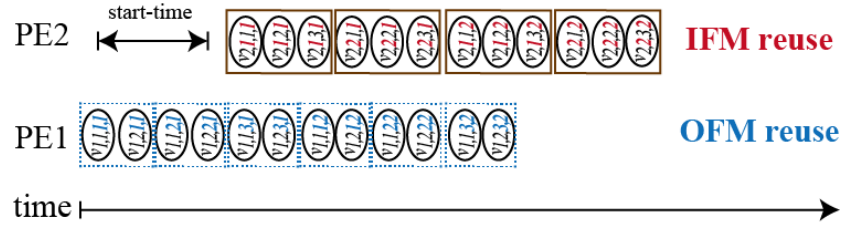
1. FPGAs with attributes including LUTs, DSPs, BRAM, etc.



2. A neural architecture with determined hyperparameters



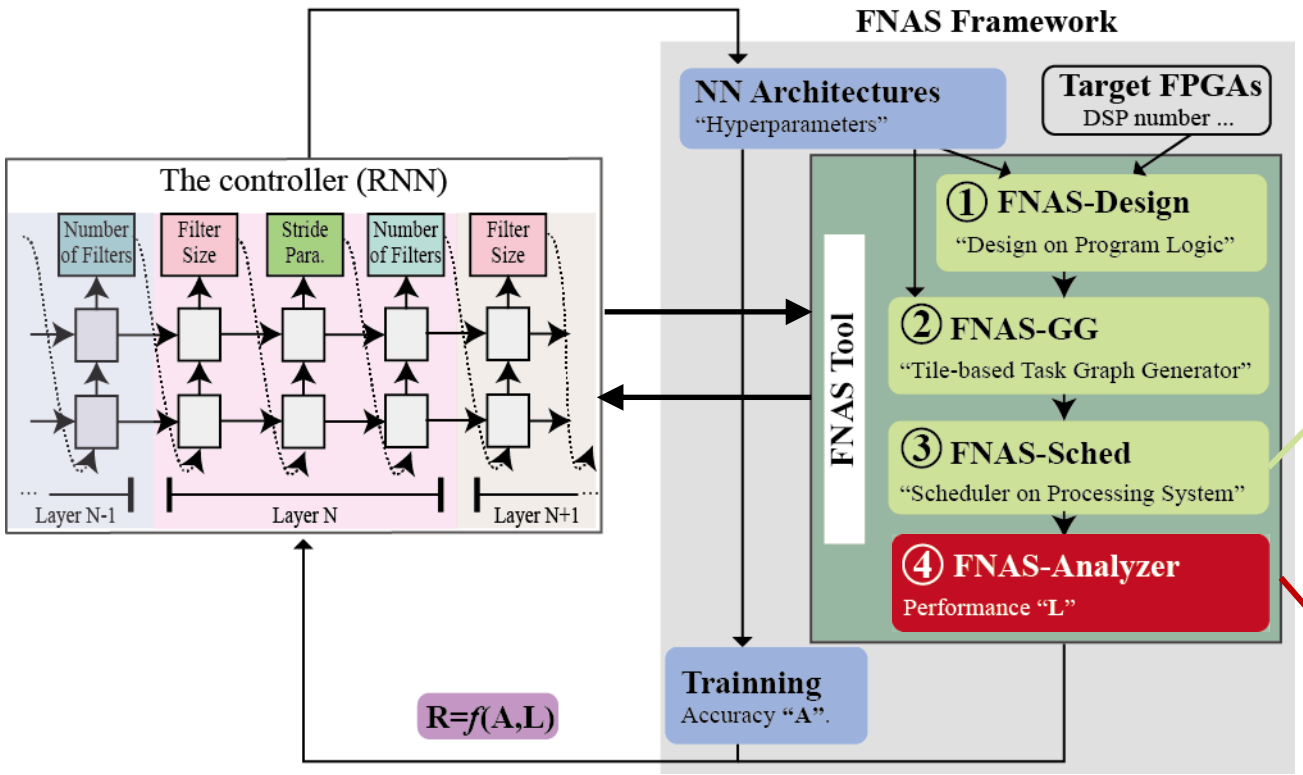
Schedule of tasks in graph on multiple accelerators



③ FNAS Sched

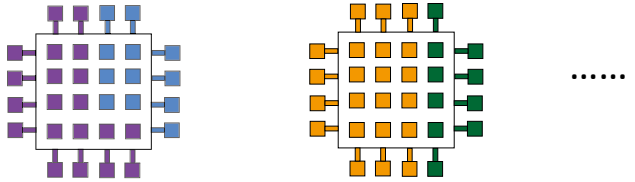
a scheduler to determine the sequence of tasks to be executed on the multiple PEs, such that the schedule length (latency) can be minimized.

FNAS: ANALYZER

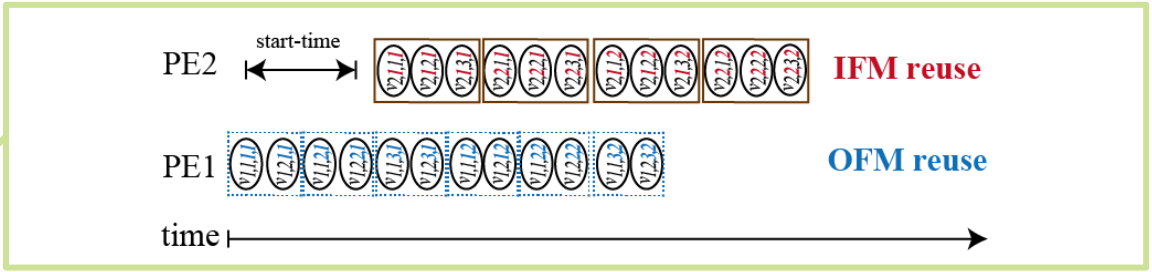


Given:

1. FPGAs with attributes including LUTs, DSPs, BRAM, etc.



2. A neural architecture with determined hyperparameters



Latency = pipeline start time + processing time

④ FNAS Analyzer

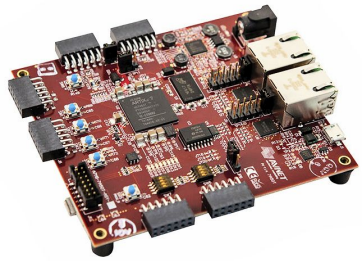
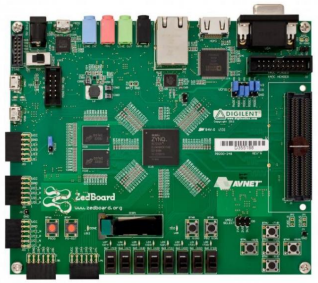
aims to compute the latency L of a neural architecture efficiently and accurately on the target FPGAs with determined schedule

- Processing time
- Start time
- Stall time

Output:

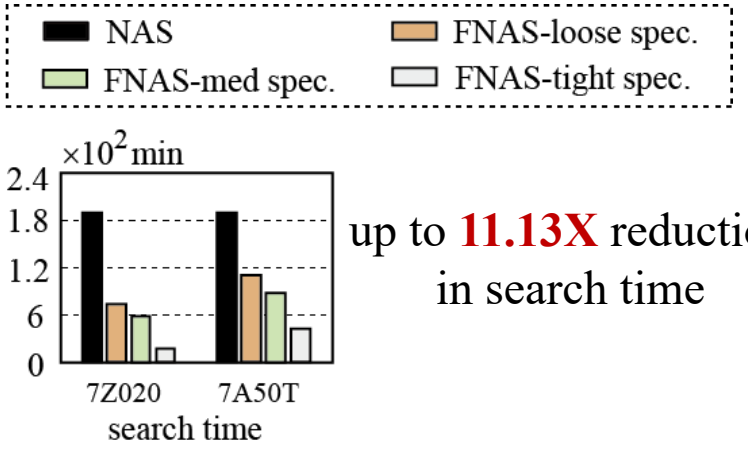
1. A tailored FPGA Design
2. The system latency

EXPERIMENTAL SETTING

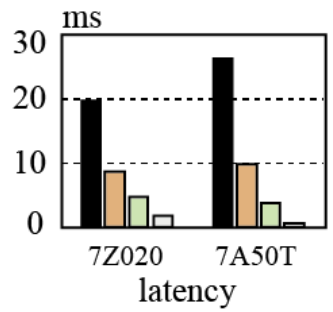
FPGAs		Xilinx 7A50T		Xilinx 7Z020
Datasets				
	MNIST	CIFAR-10	ImageNet	
NAS Search Space	Layer Num.	up to 5	up to 10	up to 15
	Filter Size	[5, 7, 14]	[1, 3, 5, 7]	[1, 3, 5, 7]
	Filter Num.	[9, 18, 36]	[24, 36, 48, 64]	[16, 32, 64, 128]
HW Search Space	Channel Tiling Para. (Tm,Tn); Row Tiling Para. (Tr); Col Tiling Para. (Tc); Schedule			
Timing Spec. (ms)		[2, 5, 10, 20]	[1.5, 2, 2.5, 10]	[2.5, 5, 7.5, 10]

Experimental Results

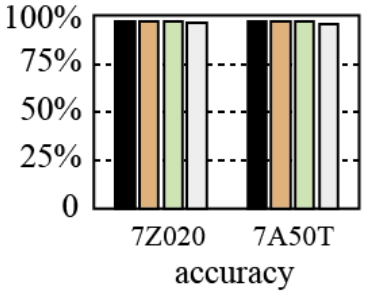
Different Hardware (MNIST)



up to **11.13X** reduction in search time

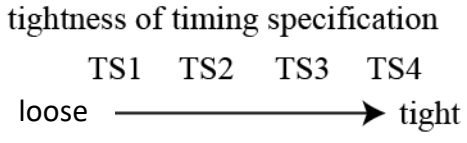


up to **7.81X** reduction in inference latency

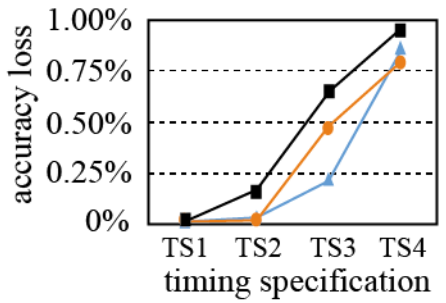


below **0.9%** loss in accuracy

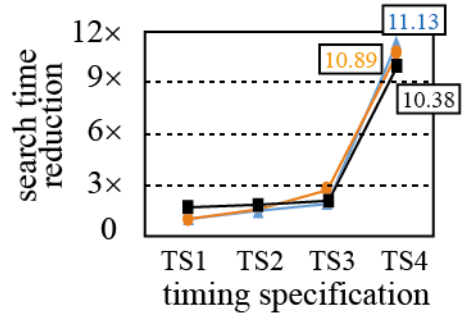
Different Datasets (7Z020)



Baseline: NAS

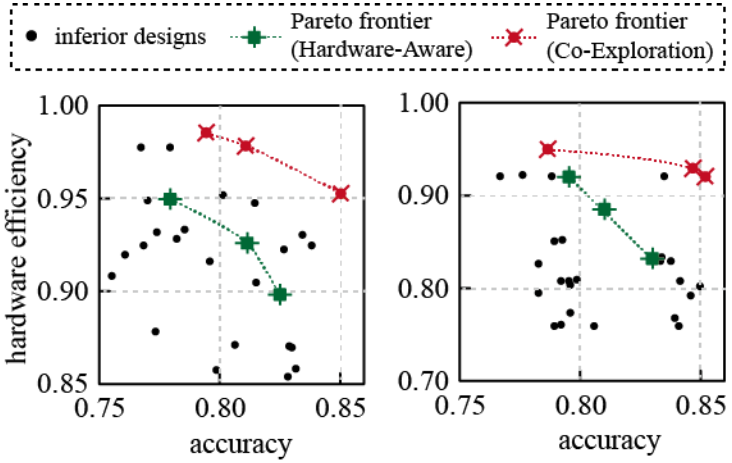


below **1%** loss in accuracy



up to **10X** reduction in inference latency

Compare to HW-Aware NAS (CIFAR-10 + 7Z020)



FNAS can significantly **push forward** the Pareto frontiers between **accuracy and efficiency** tradeoff

Experimental Results: Superior to Existing Approaches

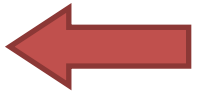
Optimizing Hardware Efficiency

Comparison the proposed Co-Exploration with Hardware-Aware NAS and Heuristic Sequential Optimization

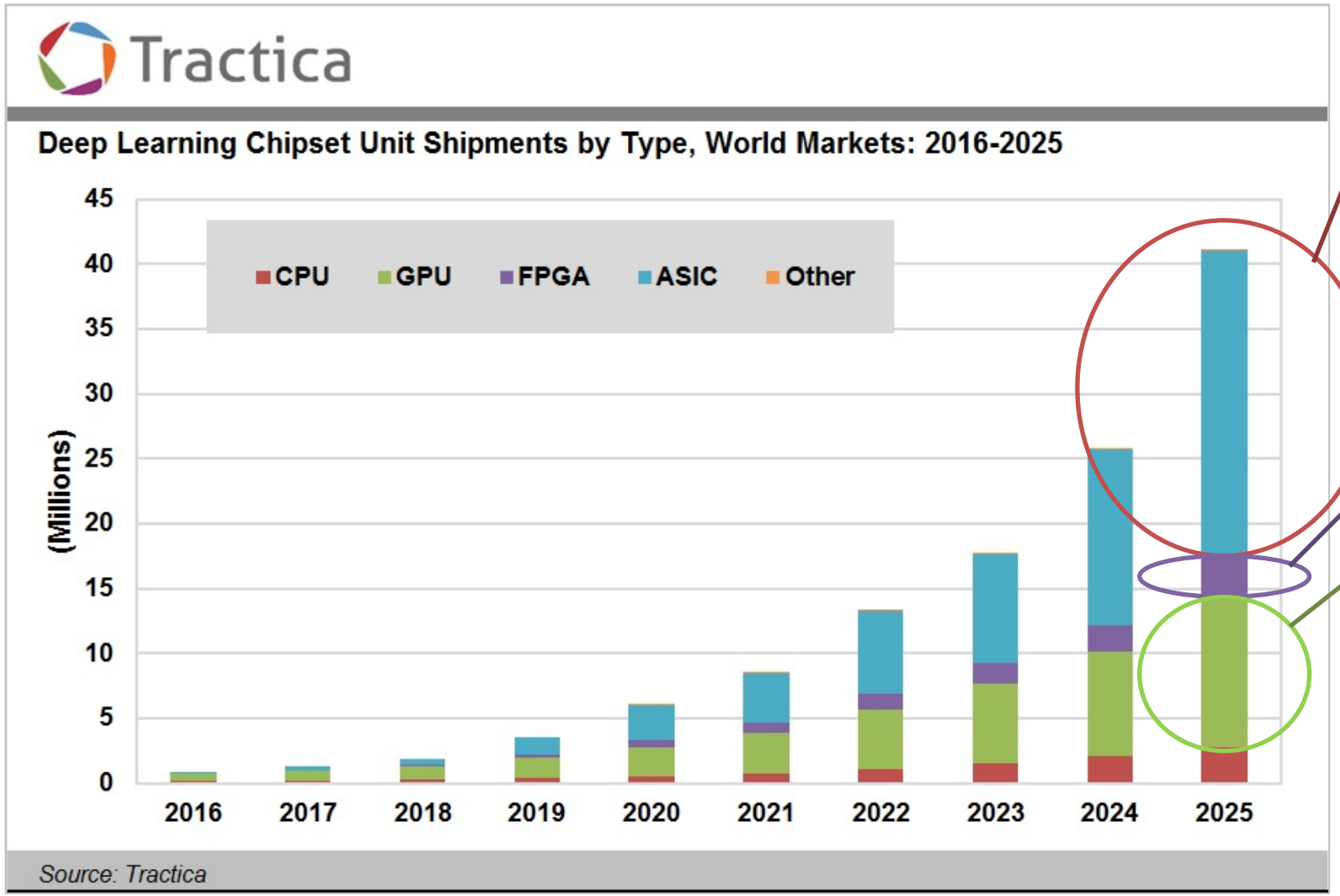
Dataset	Models	Depth	Parameters	Accuracy (Top1)	Accuracy (Top5)	Pipeline Eff.	FPS	Energy Eff. GOPS/W
CIFAR-10	Hardware-Aware NAS	13	0.53M	84.53%	-	73.27%	16.2	0.84
	Sequential Optimization	13	0.53M	84.53%	-	92.20%	29.7	1.36
	Co-Exploration (OptHW)	10	0.29M	80.18%	-	99.69%	35.5	2.55
	Co-Exploration (OptSW)	14	0.61M	85.19%	-	92.15%	35.5	1.91
ImageNet	Hardware-Aware NAS	15	0.44M	68.40%	89.84%	81.07%	6.8	0.34
	Sequential Optimization	15	0.44M	68.40%	89.84%	86.75%	10.4	0.46
	Co-Exploration (OptHW)	17	0.54M	68.00%	89.60%	96.15%	12.1	1.01
	Co-Exploration (OptSW)	15	0.48M	70.24%	90.53%	93.89%	10.5	0.74

Optimizing Network Accuracy

OUTLINE

- Hardware Design Space
- Software Design Space
- First Step: Co-Exploration of Neural Architecture and FPGAs
- Second Step: Co-Exploration of Neural Architecture and Heterogenous ASICs 
- Third Step: Co-Exploration of Neural Architecture and Scalable ASICs
- Future Work

ASIC Will Have the Largest Market



ASIC

FPGA

GPU

- **ASICs have larger market than FPGA, GPU, CPU for AI applications**
  - High Energy Efficiency
  - Low Latency
  - Small Area

Design Space of ASIC is **Huge**

Dataflow (data reuse):

- Weight stationary
- Output stationary
- Row stationary
- No reuse

y:

Communication:

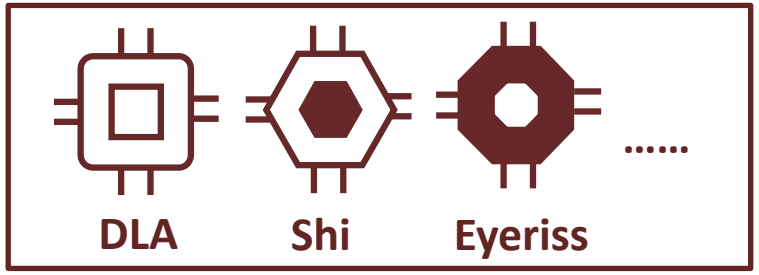
- Wormhole
- Store and forward

Motivation: Template Pool

	Existing ASIC Accelerators for Neural Network			
Dataflow →	weight stationary (WS)	output stationary (OS)	row stationary (RS)	no local reuse (NLR)
Template →	DLA (NIVIDA'17)	ShiDianNao (CAS&EPFL'15)	Eyeriss (MIT'16)	DianNao (CAS'14)
	nn-X, TPU, ……	Gupta, Peemen, ……	Eyeriss v2, ……	DaDianNao, Zhang, ……

Well-designed ASICs → No need to design ASIC from scratch!

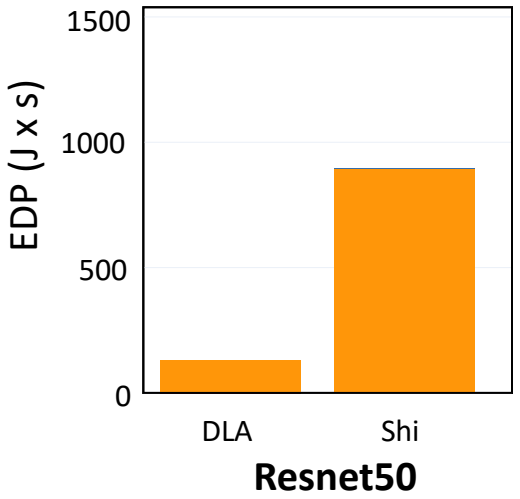
Just select the ASIC design (called Template) from the pool



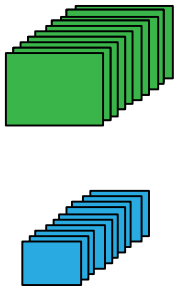
Template Pool

One template for all neural network architectures?

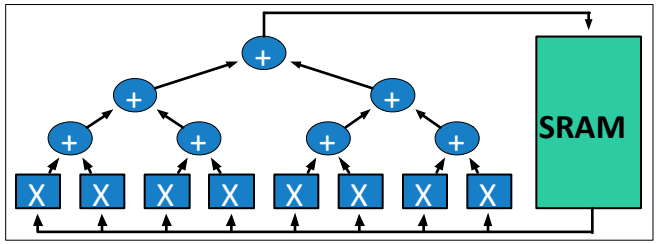
Motivation: Heterogeneous ASICs



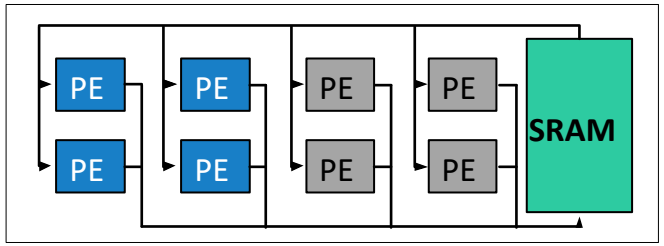
Low Resolution Deep Channels

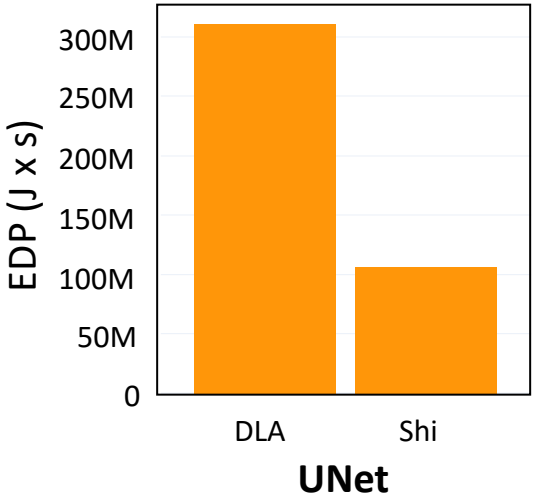


Weight Stationary Dataflow (DLA)

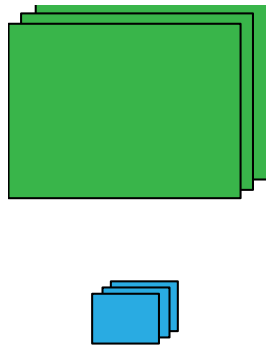


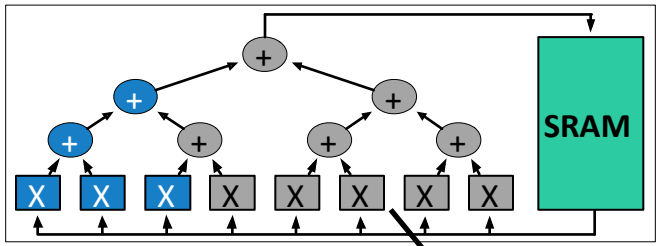
Output Stationary Dataflow (ShiDianNao)

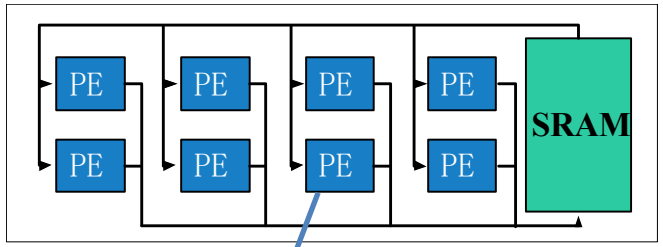




High Resolution Shallow Channels







Energy-Delay-Product
(**the Smaller the Better**)

Activation

Filter

- Design Space: Take ASIC Accelerator with 2 Sub-Accelerator as An Example

Dataflow (data reuse):

- Weight stationary
- Output stationary
- Row stationary
- No reuse

Topology:

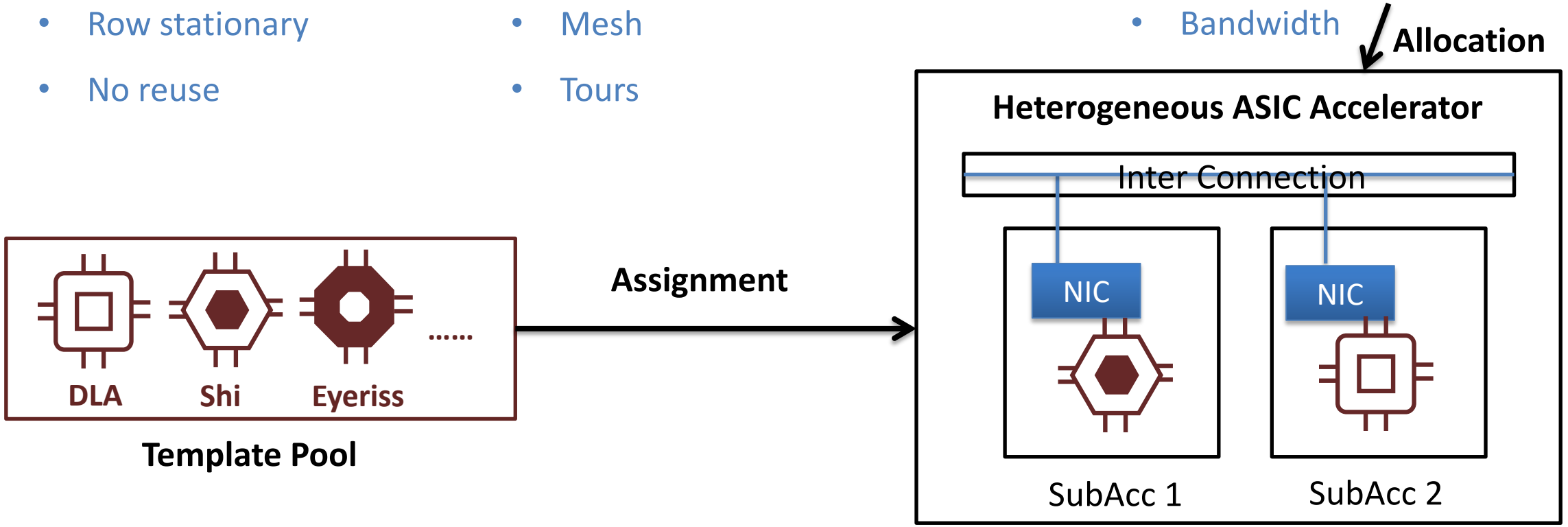
- **Bus**
- Star
- Mesh
- Tours

Communication:

- **Wormhole**
- Store and forward

Resource Allocation:

- Processing Element
- Memory
- Bandwidth



- NIC: Network Interface Controller

Problem and Challenges

Realistic Problem from

On-Device AI Group (AR/VR Glasses)

facebook

Given Design Space:

- Datasets for multiple machine learning tasks
- A set of PEs and total Bandwidth

Given Constraints:

- Latency; Power; Area.

Output:

- Neural architecture with the maximum accuracy
- ASIC chip design to satisfy hardware specifications

Challenge1: ASIC has huge design space.

For the same # of PEs, it can have

- Different topologies (Bus, NoC…)
- Different dataflows (WS, OS…)

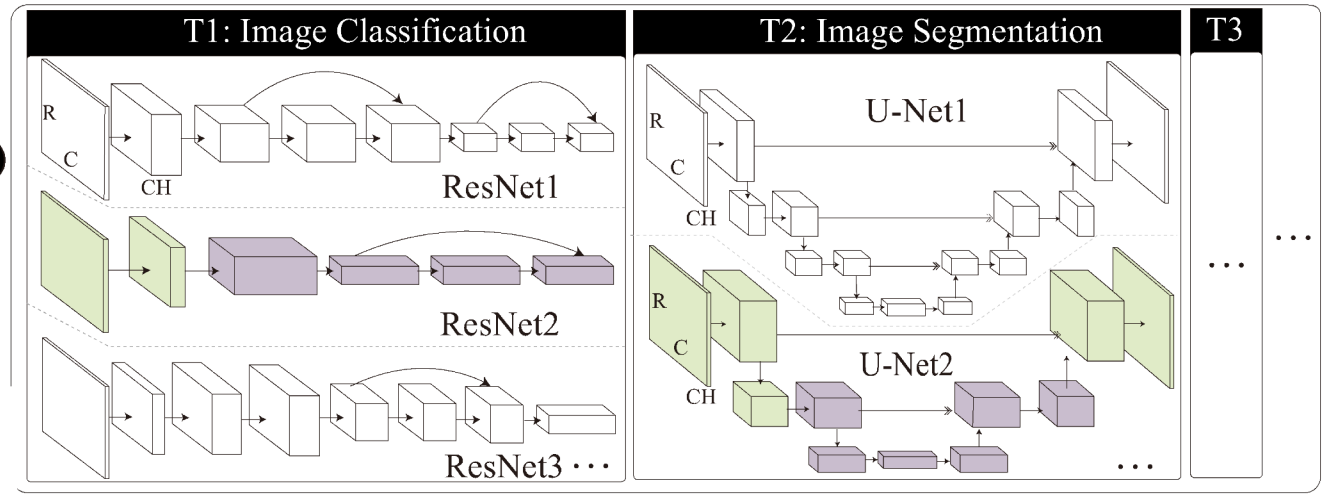
Challenge2: Multiple tasks in application

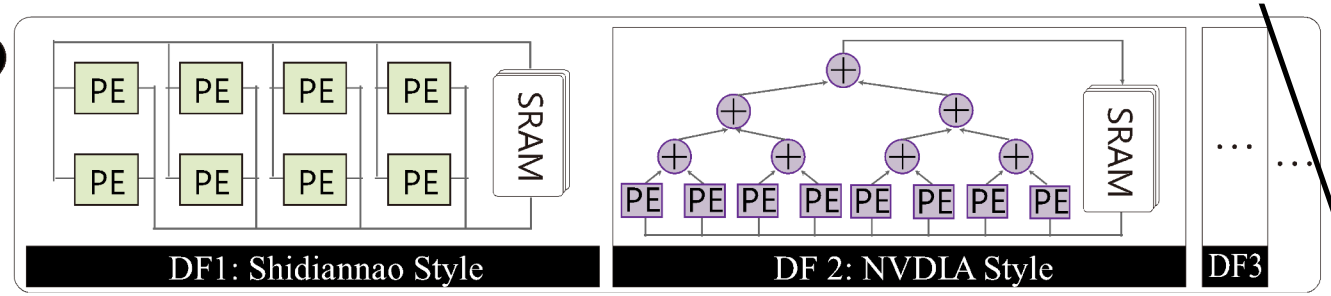
- They subject to the unified constraints (latency, power, area)

Existing NAS:

NEITHER ASICs NOR multi-tasks

Put All Together



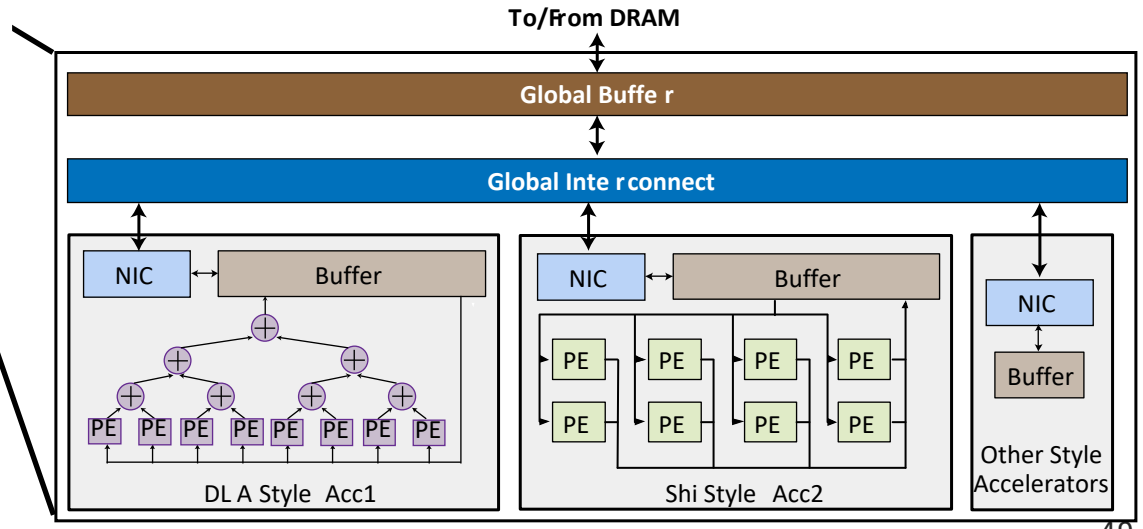


Given:

- Multiple machine learning tasks
- A pool of ASIC templates
- Unified Constraints: Latency; Power; Area.

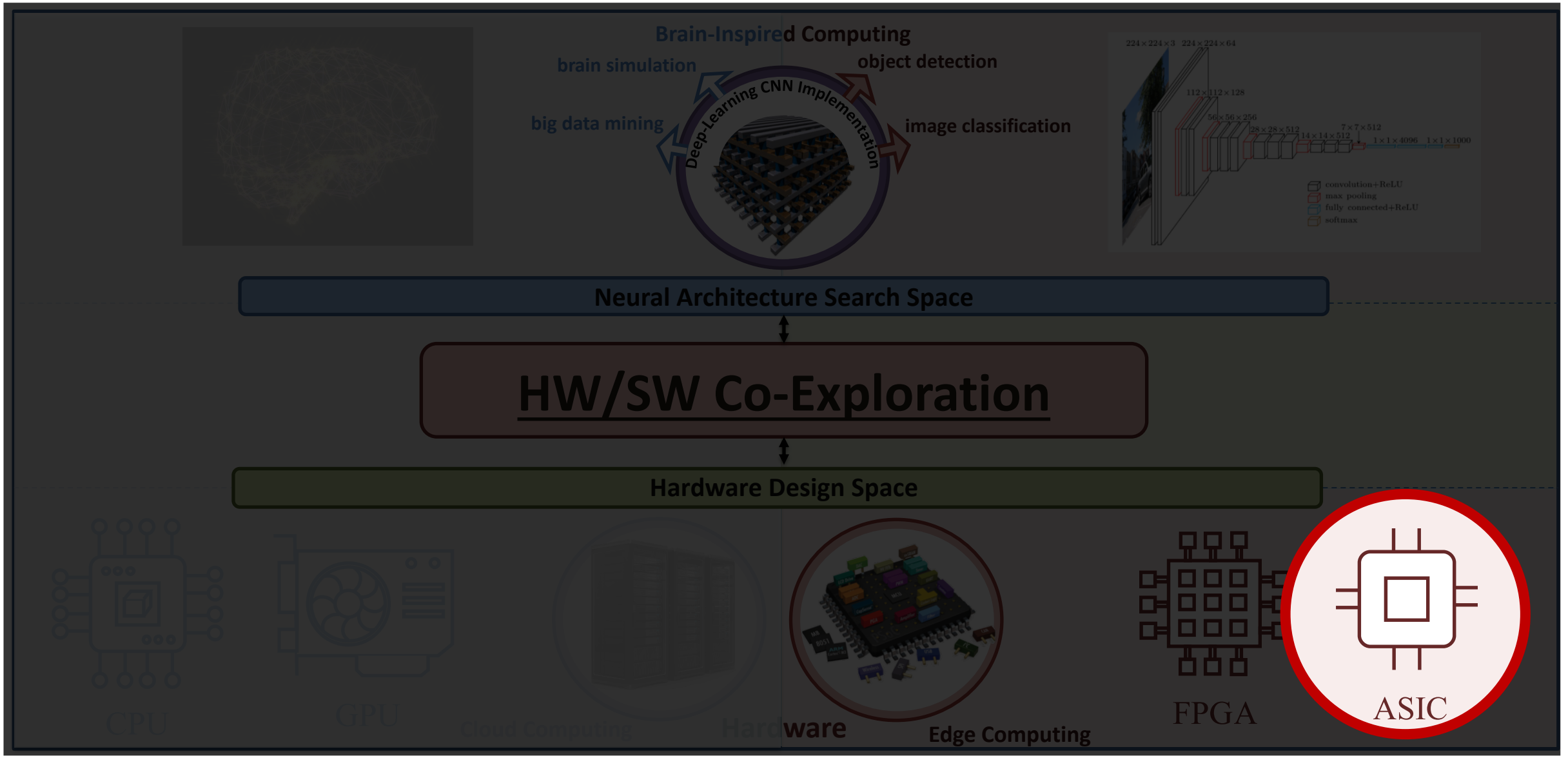
To:

- Determine neural arch. for each task (NAS)

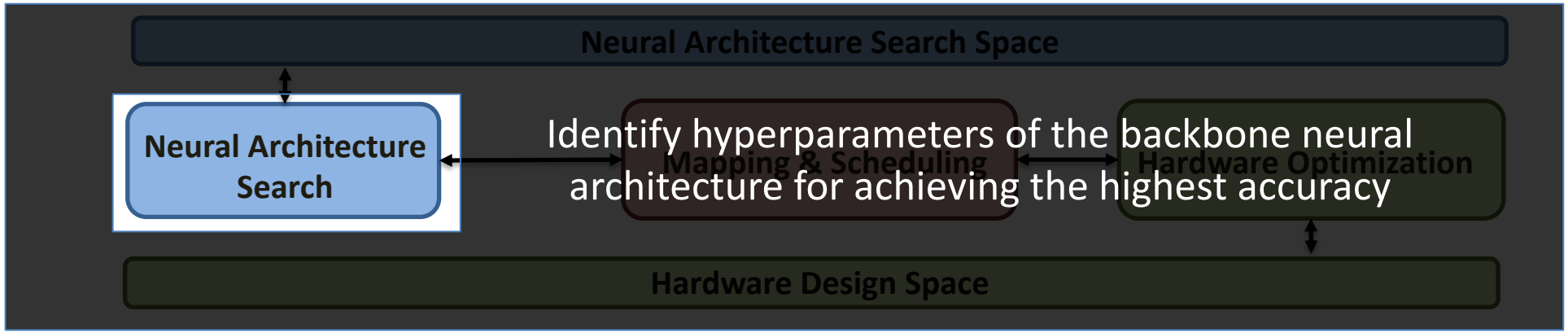


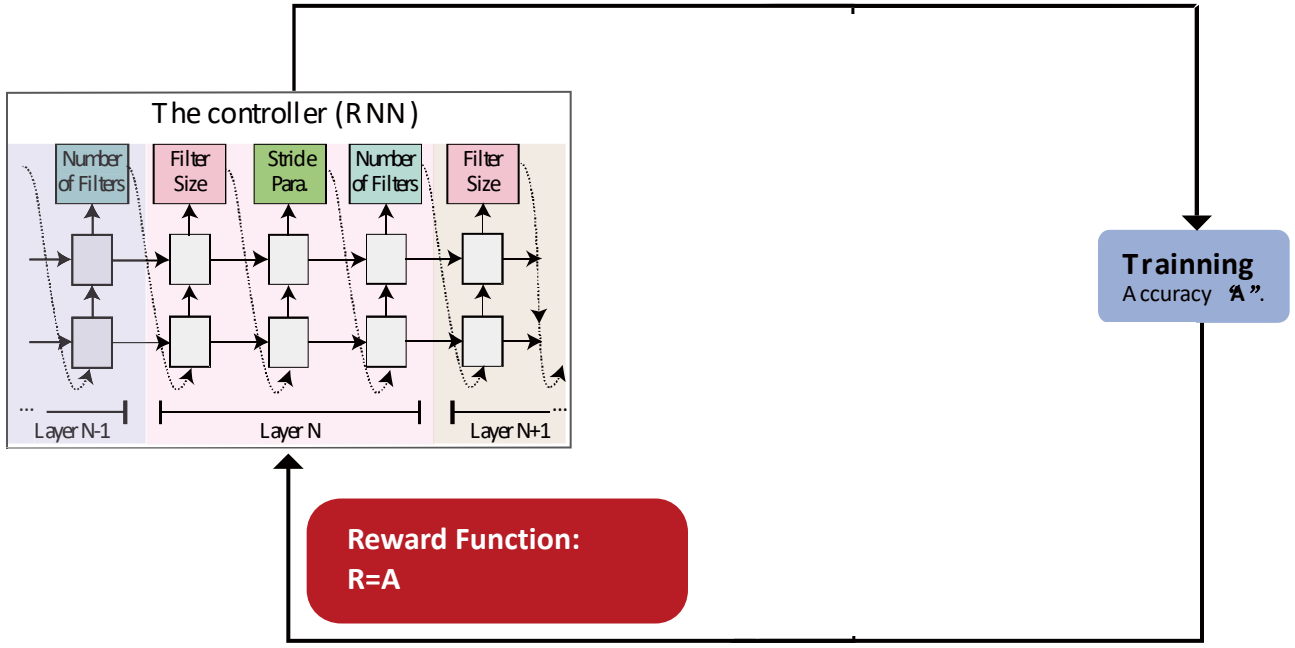
- NIC: Network Interface Controller

HW/SW Co-Exploration target ASICs

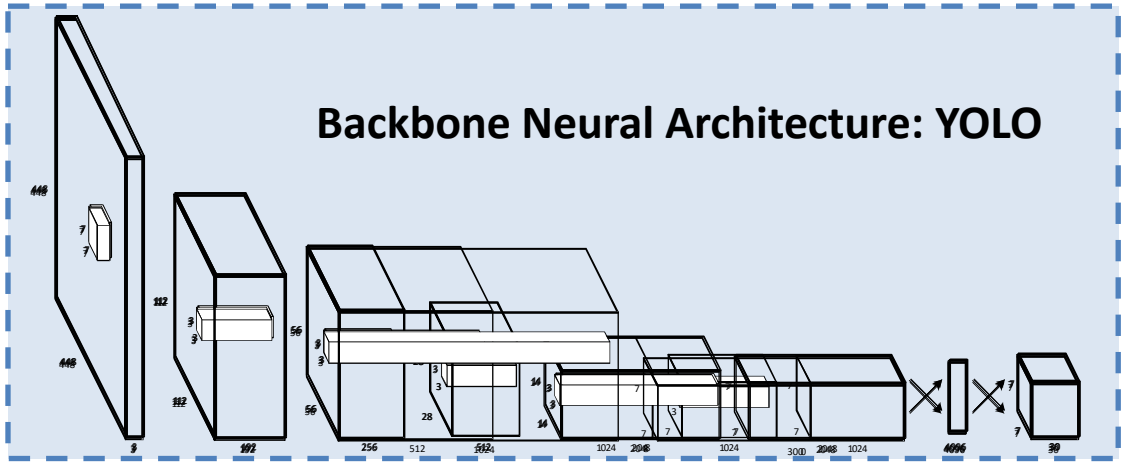


HW/SW Co-Exploration: NAS

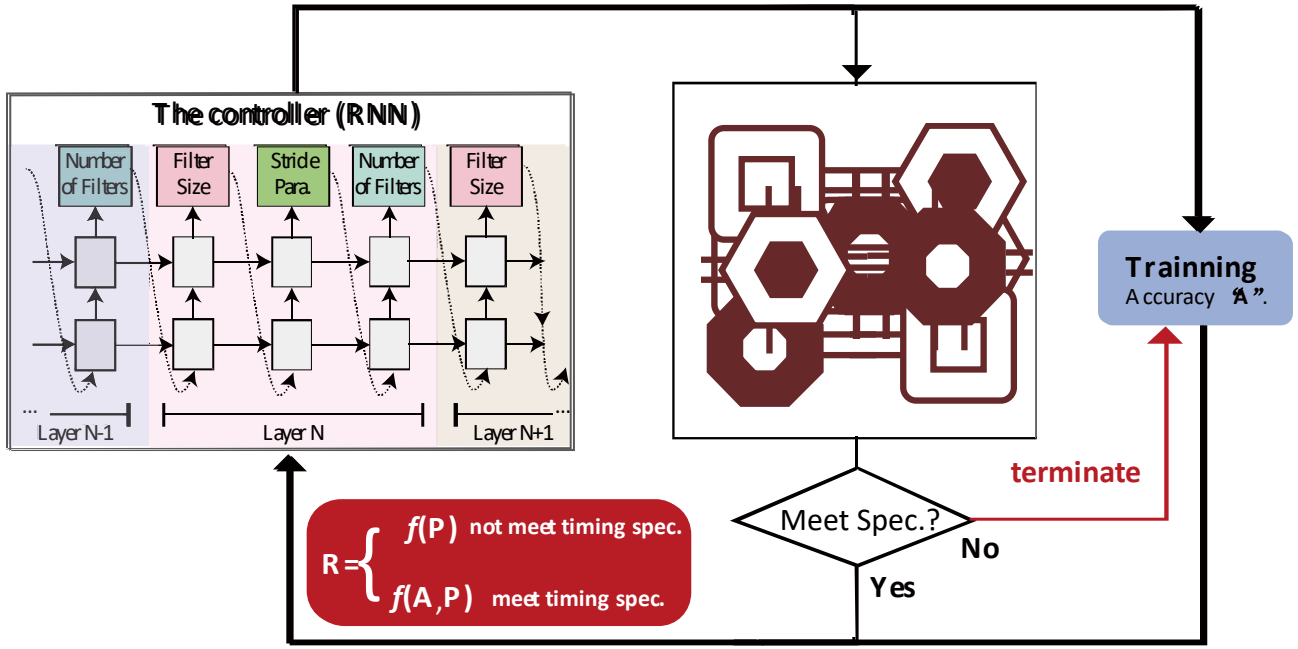




- **Apply Reinforcement Learning based NAS.**
  - Can be **any other search methods**, like evolutional algorithms, differentiable architecture search.



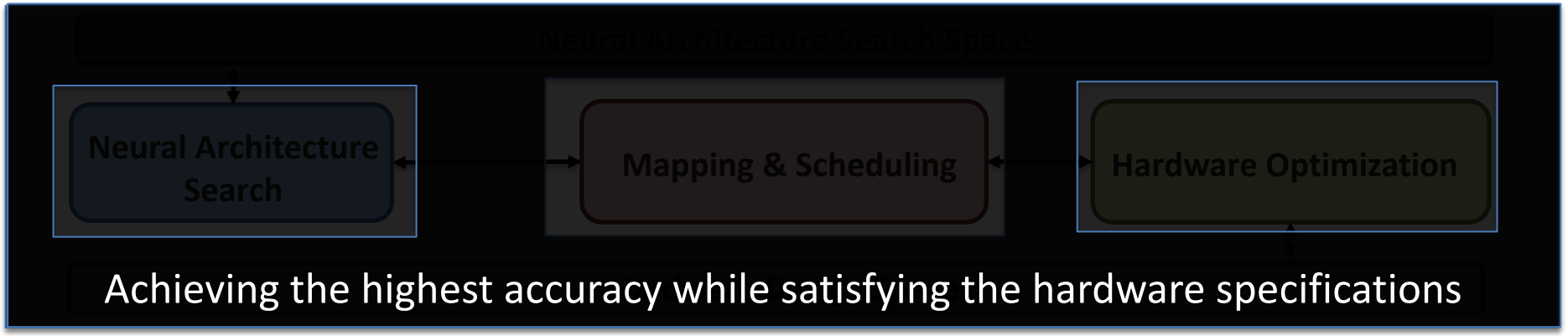
HW/SW Co-Exploration: HW Optimization

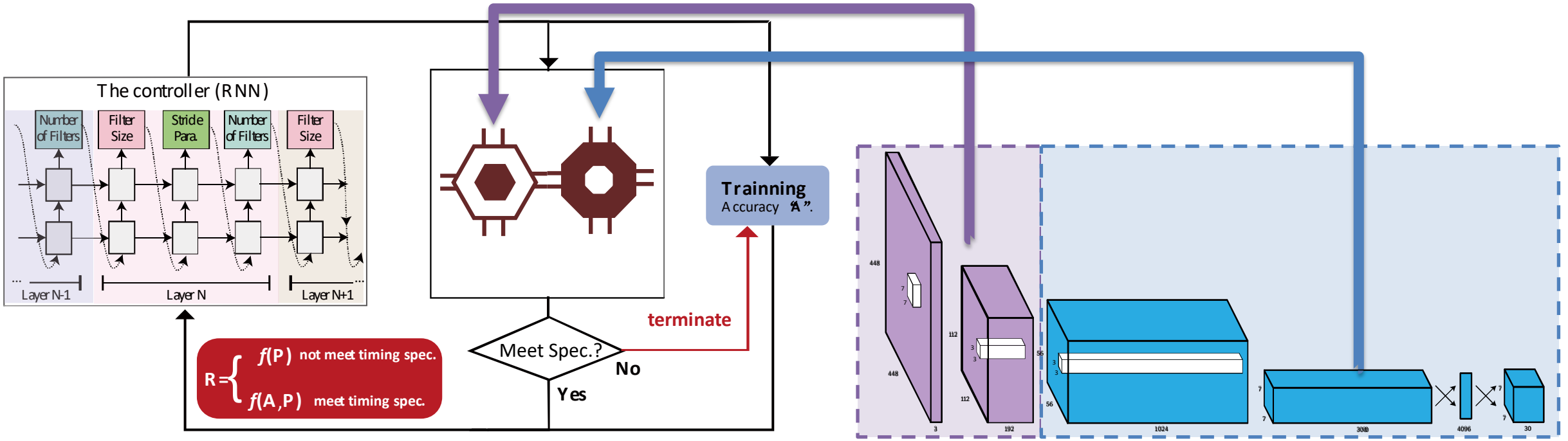


- **Add Hardware Design Module.**
  - Can terminate lengthy training if latency cannot satisfy the real-time constraints.
- **Modify Reward Function.**
  - Area, Latency, Energy and Accuracy can be simultaneously optimized.

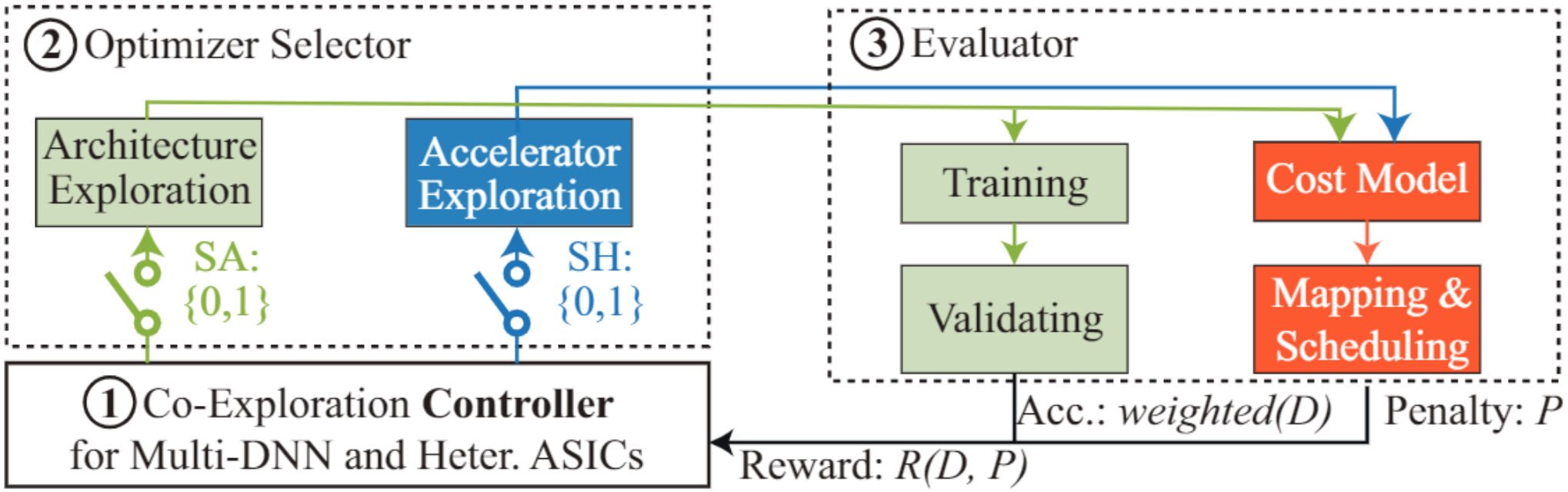
$$P = Penalty(Lat, Area, Power) = \begin{cases} 0 & Lat \le cL \quad and \quad Are \le cA \quad and \quad Energy \le cE \\ min(Lat - cL, 0) + min(Area - cA, 0) + min(Energy - CE) & otherwise \end{cases}$$

HW/SW Co-Exploration: Mapping and Scheduling





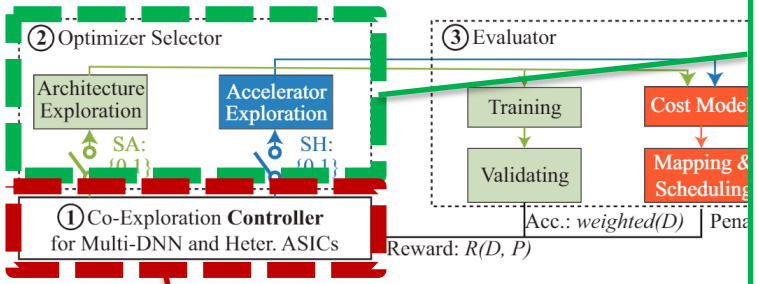
PUT ALL TOGETHER: NOVEL NASAIC FRAMEWORK



- ❖ Controller: sample NN and allocate hardware resource in each iteration
- ❖ Optimizer selector: NAS and hardware optimization
- ❖ Evaluator: generate the accuracy and hardware cost
- ❖ Finally, a reward is generated to update the controller

➡ generate solutions with high weighted accuracy & guaranteed hardware specification

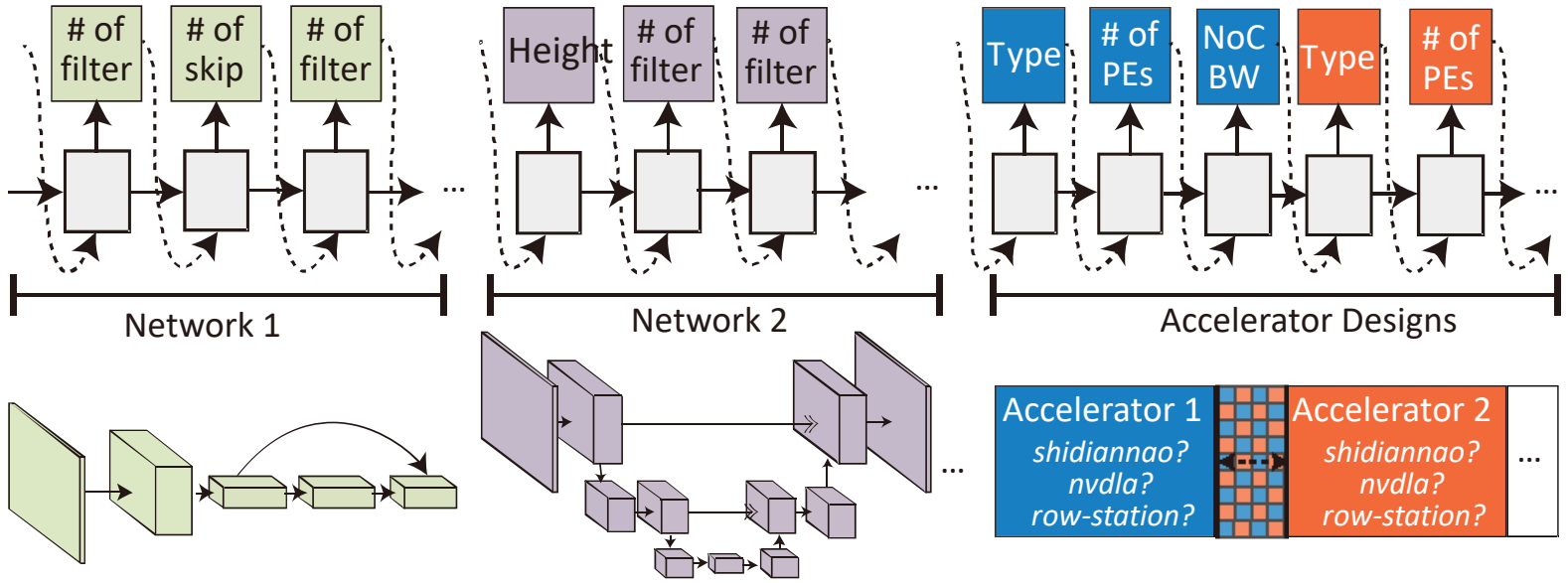
NASAIC: Controller and Selector



Selector

- SA=1; SH=0: Traditional NAS
- SA=0; SH=1: Utilize **previous neural architecture** and explore hardware space only
- SA=1; SH=1: **Co-exploration** of new architectures and hardware design

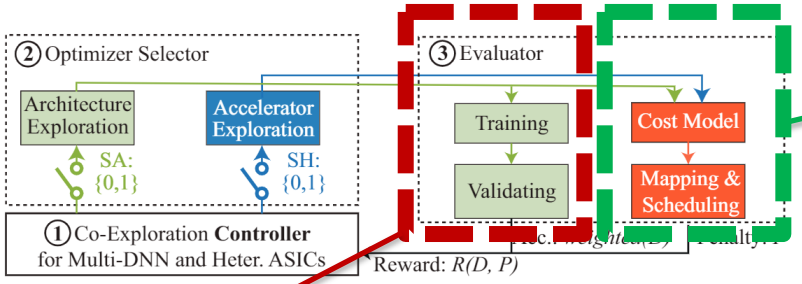
Controller



RNN: simultaneously predict multiple neural architectures

Acc. Design Param.: realize co-exploration of NN and hardware designs

NASAIC: EVALUATOR



Evaluator: neural architecture accuracy

Given:

- An **identified** neural network architecture $\boldsymbol{D_i}$ for each task
- A **held-out validation dataset** for task $\boldsymbol{T_i}$ in a total of $|\boldsymbol{T}|$ tasks

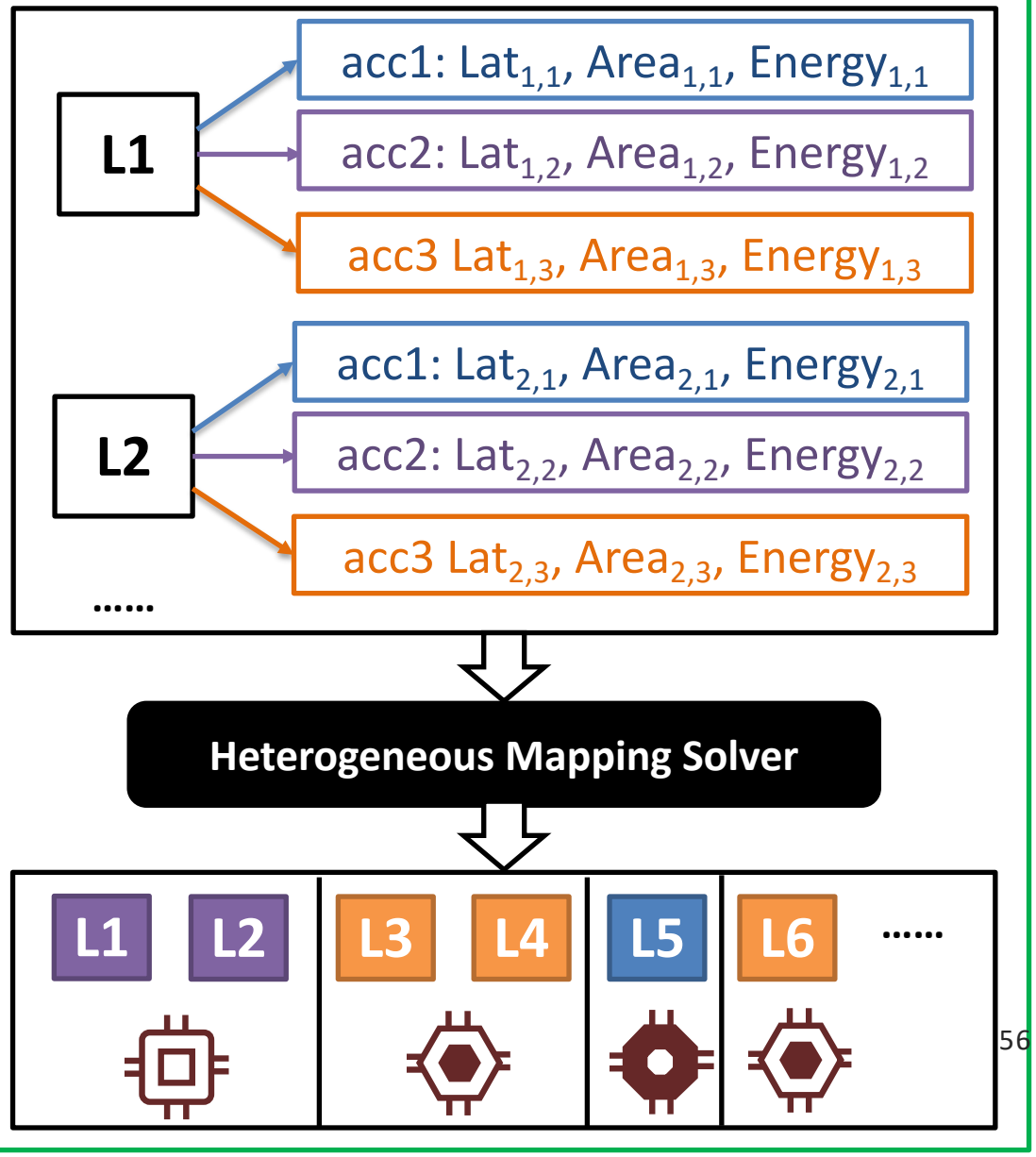
Do:

- Training and validation to obtain accuracy $\boldsymbol{acc_i}$ of $\boldsymbol{D_i}$
- Feedback **weighted accuracy**, given weight α_i of $\boldsymbol{D_i}$

$$weighted(D) = \sum_{i=1,2,\cdots,|T|} \{\alpha_i \times acc_i\}$$

Evaluator: hardware performance

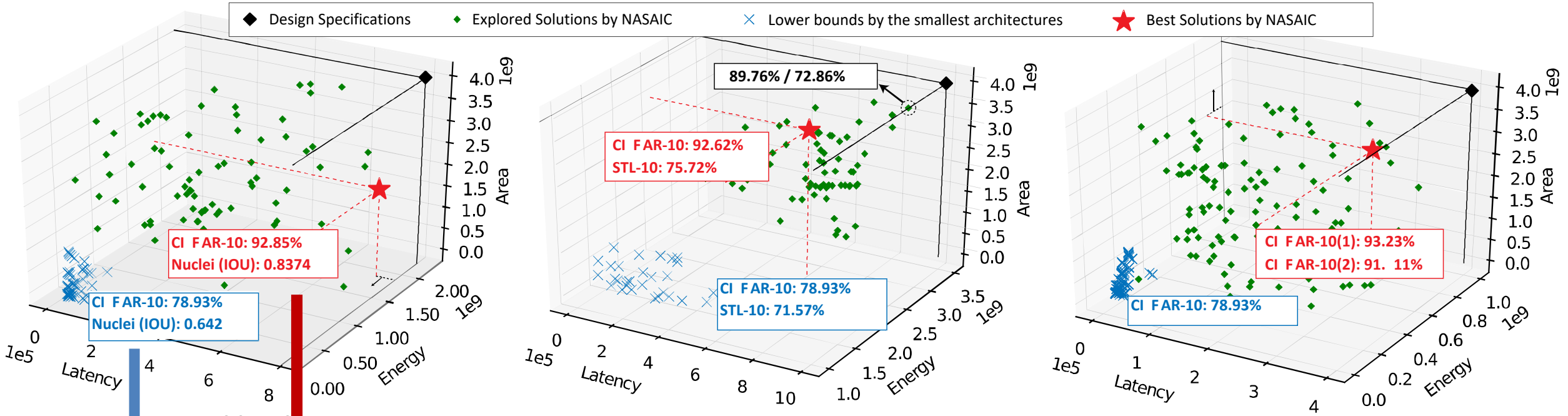
Cost Model: MAESTRO



Experiment Setup

- **Application workloads**: CIFAR-10, STL-10
- **Backbone architectures**: ResNet9, CIFAR-10, STL-10,
- **Hardware configuration**: PEs as 4096 and the maximum NoC bandwidth as 64GB/s
- **ASICNAS setting**: explore the search space for 500 episodes and explore 10 accelerator designs in each episode.

Results: Design Space Exploration



Workload W1:
CIFAR-10 (Classification)
& Nuclei (Segmentation)

Workload W2:
CIFAR-10 (Classification)
STL-10 (Classification)

Workload W3:
CIFAR-10 (Classification)
CIFAR-10 (Classification)

- Simple heuristic algorithms with minimum architecture size **cannot** provide the acceptable accuracy
- **NASAIC can guarantee**
  - ✓ All Identified architecture and hardware can meet design specifications
  - ✓ Meanwhile, achieving high accuracy

COMPARISON RESULTS ON MULTI-DATASET WORKLOADS

Sequential NAS and ASIC Design (Best Accuracy)

1. Dataflow
2. Num PE
3. Bandwidth

Sequential ASIC Design and HW-NAS

Ours

Workload			D		L/ cycles	E/ nJ	A/ μm^2
W1 CIFAR-10 Nuclei	NAS → ASIC	dla, 2112 , 48	CI I		9.45e5 ×	3.56e9 ×	4.71e9 ×
	ASIC → HW-NAS		10 ei	91.98% 83.72%	5.8e5 ◯	1.94e9 ◯	3.82e9 ◯
	NASAIC	dla, 576 , 56 792 , 8	CI FAR-10 Nuclei	**92.85%** **83.74%**	7.77e5 ◯	1.43e9 ◯	2.03e9 ◯
W2 CIFAR-10 STL-10	NAS → ASIC	368 , 56 shi, 1728 , 8	CI FAR-10 STL-10	94.17% 76.50%	9.31e5 ◯	3.55e9 ×	4.83e9 ×
	ASIC → HW-NAS	dla, 2112 , 24 shi, 1536 , 40	CI FAR-10 STL-10	92.53% 72.07.%	9.69e5 ◯	2.90e9 ◯	3.86e9 ◯
	NASAIC	dla, 2112 , 40 shi, 1184 , 24	CI FAR-10 STL-10	**92.62%** **75.72%**	6.48e5 ◯	2.50e9 ◯	3.34e9 ◯

×: Violate the design specs. ◯: Meet the design specs.

NAS --> ASIC:

- Highest accuracy
- Cannot meet design spec.

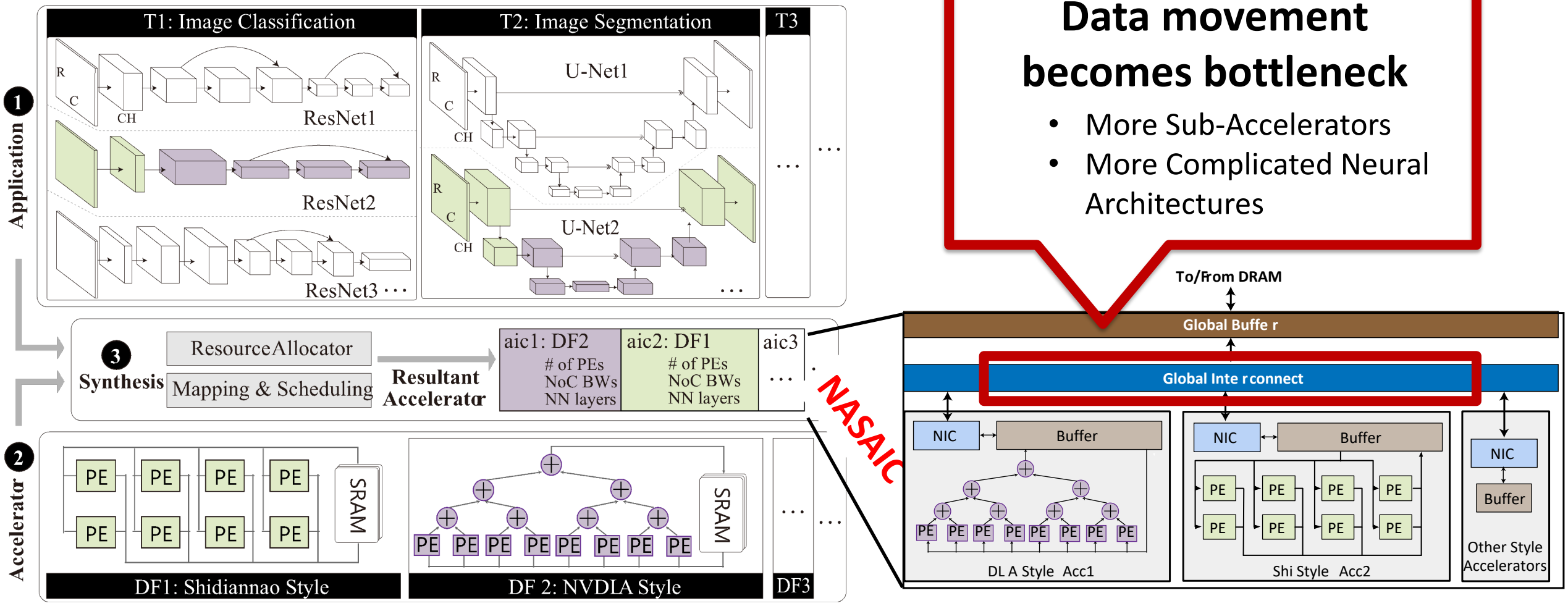
ASIC --> HW-NAS:

- Hardware is not optimized
- Accuracy is low

NASAIC:

✓ Best tradeoff between accuracy and hardware efficiency

Problem and Challenges

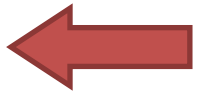


Data movement becomes bottleneck

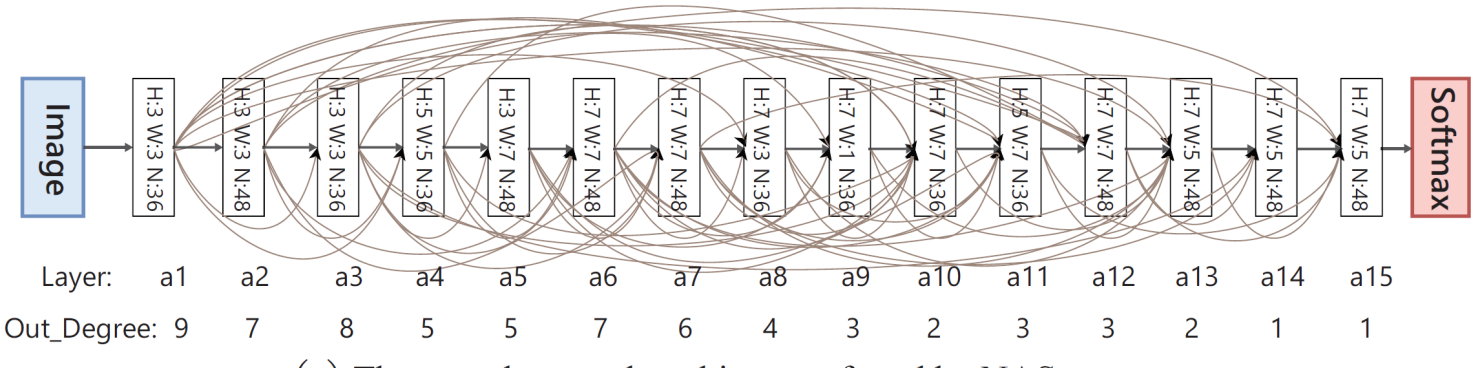
- More Sub-Accelerators
- More Complicated Neural Architectures

- NIC: Network Interface Controller

OUTLINE

- Hardware Design Space
- Software Design Space
- First Step: Co-Exploration of Neural Architecture and FPGAs
- Second Step: Co-Exploration of Neural Architecture and Heterogenous ASICs
- Third Step: Co-Exploration of Neural Architecture and Scalable ASICs 
- Future Work

Motivation: Scalable Network-on-Chip (NoC) for NN



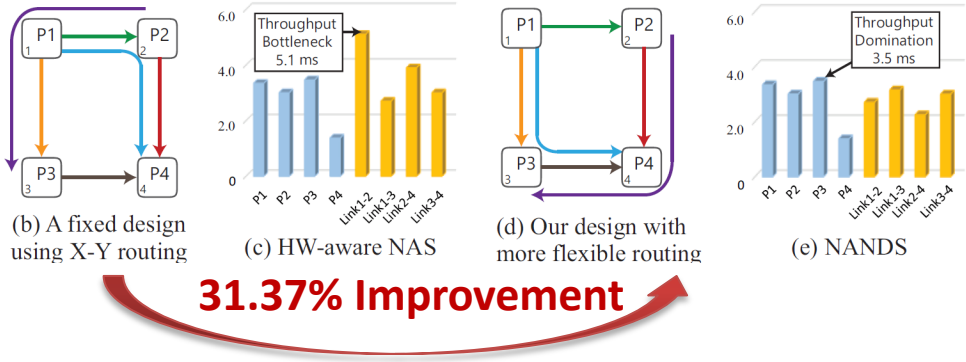
(a) The neural network architecture found by NAS

Operation Time \ Platforms	Single Processing Element	4 Processing Elements	
		Bus Interconnection	2-D Mesh NoC
Computation (ms)	12.4	3.4	3.4
Data transmission (ms)	–	14.7	6.2

(b) The timing performance of network implementations on different platforms

Observations:

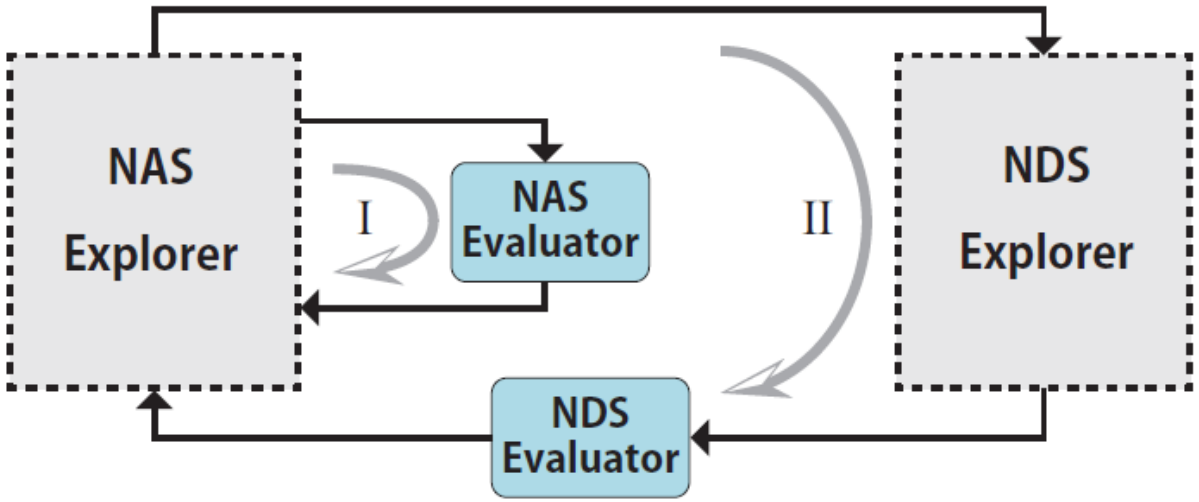
- More PEs:
  better performance
- Communication:
  performance bottleneck
- Fixed design:
  lower performance



(b) A fixed design using X-Y routing (c) HW-aware NAS (d) Our design with more flexible routing (e) NANDS

REF: Zoph, Barret, and Quoc V. Le. "Neural architecture search with reinforcement learning." *ICLR 2017*

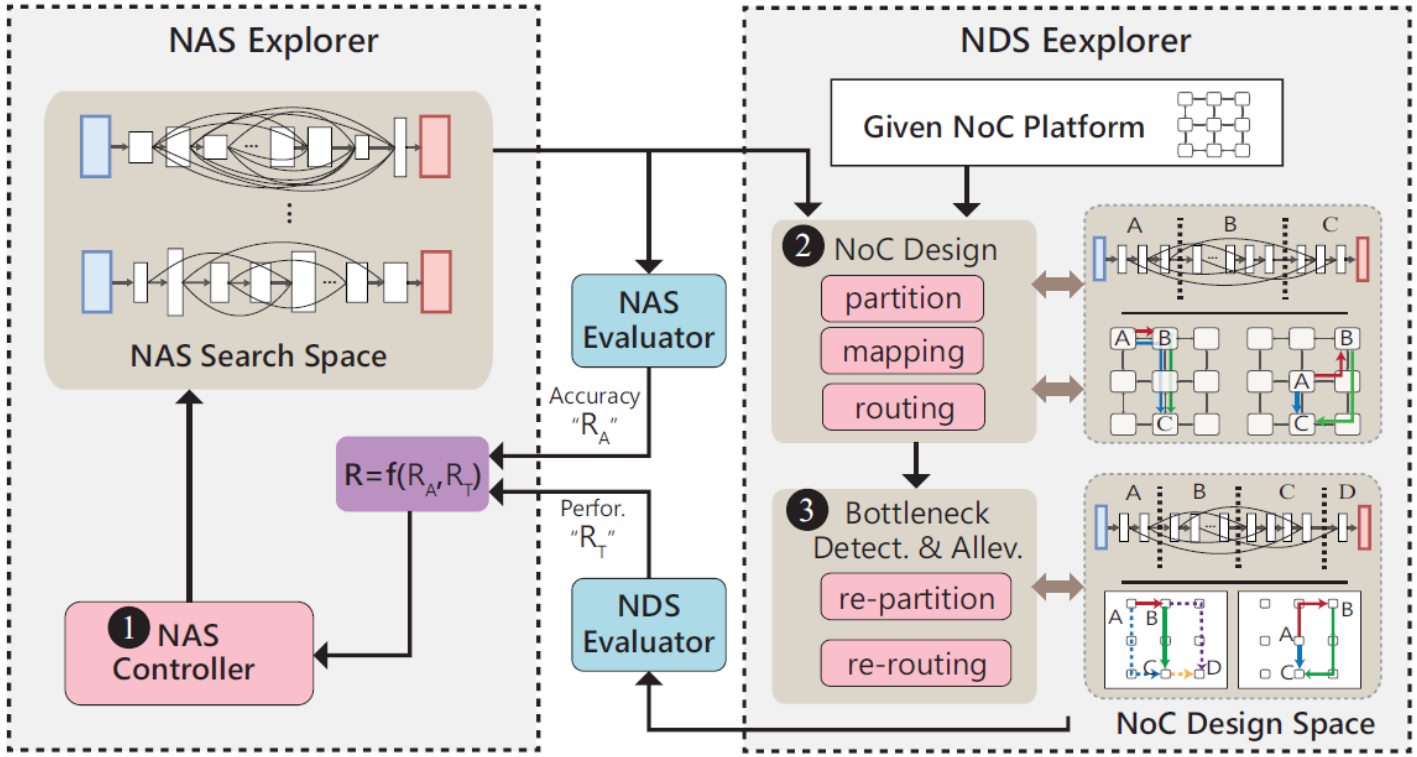
NANDS: Multi-phase Framework



Two exploration loops in NANDS:

- Loop I: Neural Architecture Search.
- Loop II: Automatic Hardware Design.

NANDS: Multi-phase Framework



① NAS Controller: predict hyperparameters to generate the child neural networks

② NoC Design: generate hardware design (e.g., partition, mapping and routing) for the input network on a given NoC

③ Bottleneck Detection and Alleviation: maximize the throughput of NoC.

NANDS: Multi-phase Framework

- Three kinds of throughput (TP) bottlenecks

  - B1: TP is determined by a processing element;
  - B2: TP is determined by a NoC link, and the link is occupied by only one data transmission path;
  - B3: TP is determined by a NoC link, where there are multiple routing paths will go through

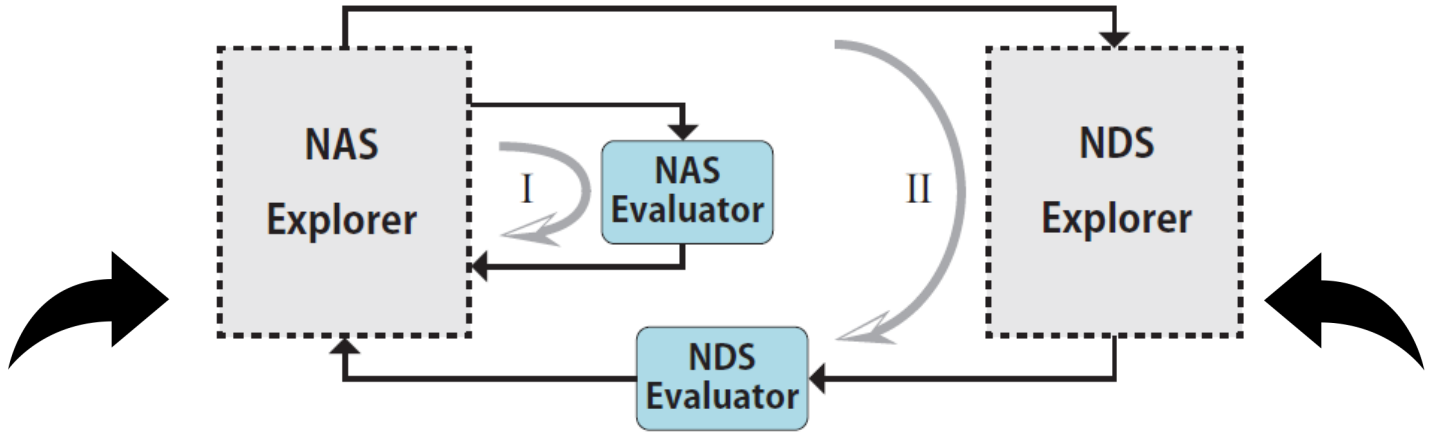
Algorithm 1 Bottleneck Alleviation

Input: (1) NoC with PE and $LINK$, (2) TP, (3) Bottleneck type, (4) latency function L, (5) partition P, (6) mapping M: p to pe, (7) routine path R.
Output: A NoC design.

if Bottleneck is B1, and TP is determined by pe_k:
    Get partition p_i, s.t., $M(p_i) = pe_k$;
    if p_i has only 1 layer:
        Cannot remove the bottleneck and terminate;
    else if NoC has available processing element:
        Partition $p_i \rightarrow p_{i1} + p_{i2}$ to minimize their max latency;
    else:
        Find p_i's neighbor p_k with the minimum latency;
        Move layers in p_i to p_k to minimize $max(L_{M(p_i)}, L_{M(P_k)})$;
else if Bottleneck is B2, and the only routing path is $pe_i \rightarrow pe_j$:
    Merge partitions on pe_i and pe_j to hide communication;
else if Bottleneck is B3, and the link is lk:
    Obtain $pe_i \rightarrow pe_j$ passing lk with the maximum data volume;
    Re-routing $pe_i \rightarrow pe_j$ to detour at lk;

Experiment Setup

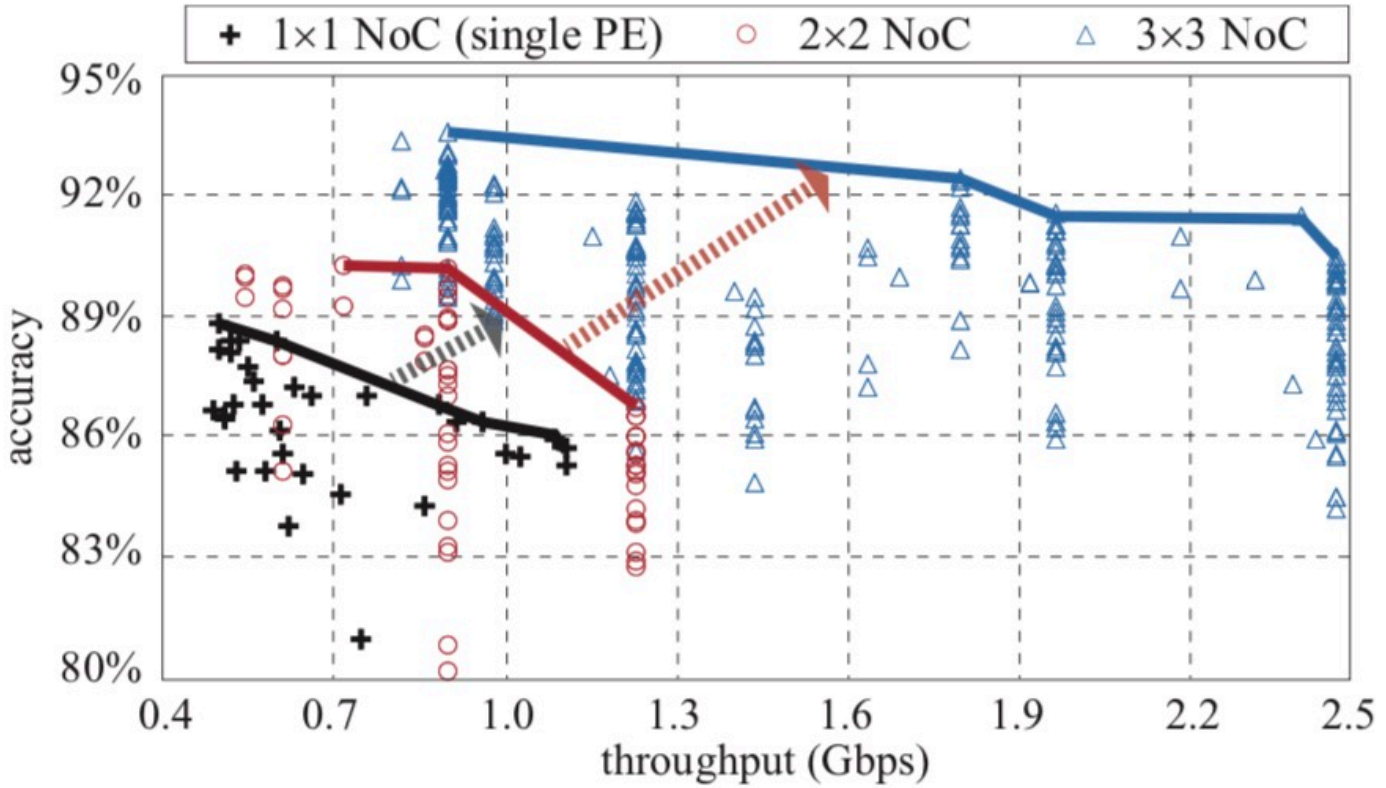
❖ **Datasets**: CIFAR-10, CIFAR-100, and STL-10



❖ **NAS Space**: ResNet as the backbone

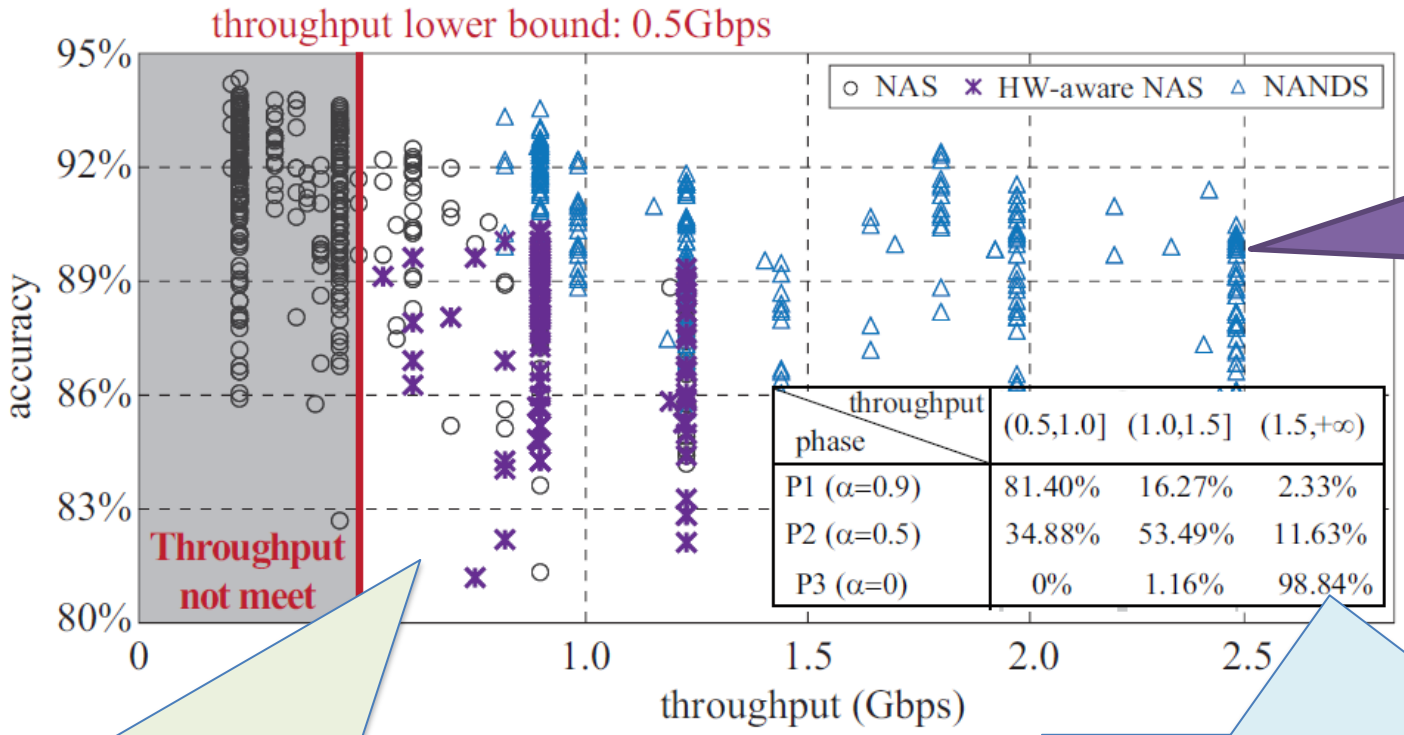
❖ **NDS Space**: 2 × 2 and 3 × 3 2-D Mesh NoCs

Experimental Results



Pareto frontiers of accuracy-throughput tradeoffs captured by NANDS on CIFAR-10 can be significantly pushed forward with increasing NoC size.

Experimental Results



phase \ throughput	(0.5,1.0]	(1.0,1.5]	(1.5,+∞)
P1 (α=0.9)	81.40%	16.27%	2.33%
P2 (α=0.5)	34.88%	53.49%	11.63%
P3 (α=0)	0%	1.16%	98.84%

NANDS significantly pushes forward Pareto frontiers, against HW-aware NAS

- Conventional NAS cannot guarantee the timing performance
- HW-aware NAS cannot find valid solutions

NANDS can guide the controller to make a better tradeoff between the accuracy and throughput.

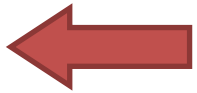
EXPERIMENTAL RESULTS

TABLE I
COMPARISON OF THE SEARCH TIME BETWEEN PURE NDS, NAS, HW-AWARE NAS, AND NANDS, ON THREE COMMON DATASETS

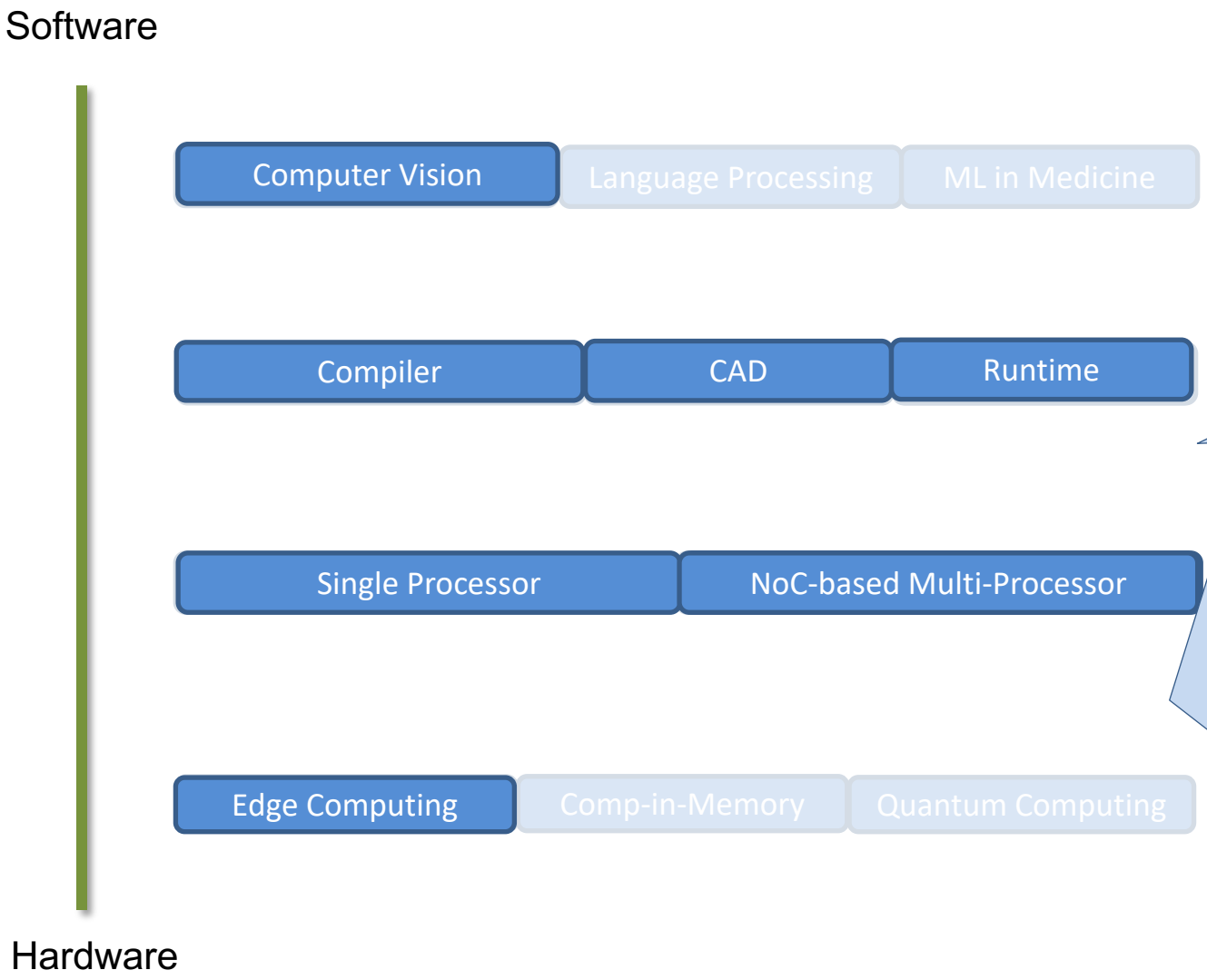
Dataset	Spec. (Gbps)	Models	Arch. Property			Accuracy		Throughput			Elapsed Time	
			Depth	Para. ($\times 10^6$)	MACs (GOP)	(%)	degr.	(Gbps)	Sat.	impr.	(minute)	impr.
CIFAR-10	0.50	NAS	9	0.89	0.70	94.41%	0.00%	0.22	×	baseline	1115	baseline
		HW-Aware NAS	8	0.19	0.05	90.95%	-3.46%	0.66	✓	2.94×	164	6.80×
		NANDS (Opt TP)	**8**	**0.20**	**0.06**	**91.58%**	**-2.83%**	**2.40**	✓	**10.66×**	**361**	**3.09×**
		NANDS (Opt Acc.)	**10**	**0.40**	**0.21**	**93.59%**	**-0.82%**	**0.90**	✓	**4.00×**		
CIFAR-100	0.45	NAS	12	1.04	1.02	76.58%	0.00%	0.22	×	baseline	1863	baseline
		HW-Aware NAS	8	0.19	0.07	71.43%	-5.15%	0.28*	×	1.25×	246	7.57×
		NANDS (Opt TP)	**8**	**0.25**	**0.15**	**72.22%**	**-4.36%**	**0.90**	✓	4.00×	**594**	**3.14×**
		NANDS (Opt Acc.)	**12**	**0.63**	**0.46**	**75.58%**	**-1.00%**	**0.45**	✓	2.00×		
STL-10	0.6	NAS	11	2.95	2.13	76.45%	0.00%	0.45	×	baseline	2928	baseline
		HW-Aware NAS	12	1.70	0.50	74.25%	-2.20%	0.61	✓	1.25×	402	7.28×
		NANDS (Opt TP)	**11**	**2.02**	**1.02**	**75.83%**	**-0.62%**	1.07	✓	2.37×	**1059**	**2.76×**
		NANDS (Opt Acc.)	**13**	**2.65**	**1.45**	**76.45%**	**0.00%**	0.60	✓	1.32×		

"*": relax spec., HW-aware NAS cannot guarantee throughput of 0.45Gbps.

OUTLINE

- Hardware Design Space
- Software Design Space
- First Step: Co-Exploration of Neural Architecture and FPGAs
- Second Step: Co-Exploration of Neural Architecture and Heterogenous ASICs
- Third Step: Co-Exploration of Neural Architecture and Scalable ASICs
- Future Work 

➢ Emerging Hardware & Application Create New Opportunities!



Co-Explore NAS & Edge Computing (Postdoc work):

- ACM TECS'19
- CODES+ISSS'19 (BPN)
- FPGA'19
- DAC'19 (BPN)
- IEEE TCAD'18
- CASES'18
- DAC'20
- ASP-DAC'19 (BPN)

Computing Architecture (Ph.D. work):

- ASP-DAC '19 (BPN)
- IEEE TC '18
- IEEE TCAD '18
- CODES+ISSS '18
- FGCS '17
- IEEE TPDS '17
- ASP-DAC '17
- DAC '17
- EMSOFT'17
- IEEE TVLSI '16
- ASP-DAC'16 (BPN)
- HPCC'15
- ISVLSI'14
- RTCSA '14

Expand the HW-SW Co-Exploration

Software

Computer Vision | Language Processing | ML in Medicine

Compiler | CAD | Runtime

Single Processor | NoC-based Multi-Processor

Edge Computing | Compute in Memory | Quantum Computing

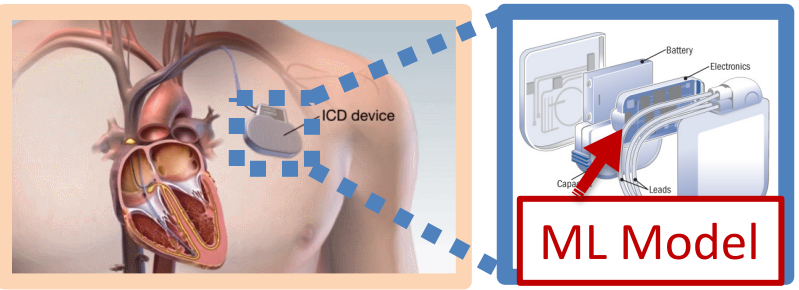
Hardware

Topics:

- Real-Time Voice Language Translation in AI Glasses
- Combine Vision and Audio

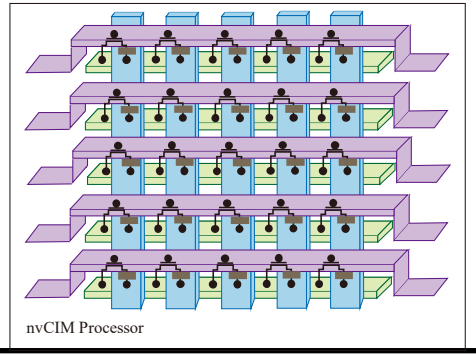
Topics:

- Squeeze ML model to ICD devices



Topics:

- Migrate bottlenecks on data movement
- Variation-aware neural architecture search



Co-Explore Quantum Computing

Software

Machine Learning Applications

Compiler | CAD | Runtime

Single Processor | NoC-based Multi-Processor

Edge Computing | Comp-in-Memory | Quantum Computing

Hardware

Topics:

- Co-Explore Neural Network with Quantum computer

IBM

IBM& University of Notre Dame Quantum program

(access IBM Q 53 qbits)